HITACHI
Inspire the Next

スケーラブルデータベースサーバ

# HiRDB Version 8 システム導入・設計ガイド（Windows(R) 用）

解説・手引・操作書

3020-6-352-42

## ■対象製品

●**適用** OS：Windows 2000，Windows Server 2003 x64 Editions，Windows Server 2008 R2，Windows Server 2008 (x64)，Windows XP x64 Edition，Windows Vista Ultimate (x64)，Windows Vista Business (x64)，Windows Vista Enterprise (x64)，Windows 7 Professional (x64)，Windows 7 Enterprise (x64)，Windows 7 Ultimate (x64)

P-2462-7184 HiRDB/Single Server Version 8 08-51

P-2462-7384 HiRDB/Parallel Server Version 8 08-51

P-2462-7H84 HiRDB Non Recover Front End Server Version 8 08-00

P-2462-7J84 HiRDB Advanced High Availability Version 8 08-00

P-2462-7K84 HiRDB Advanced Partitioning Option Version 8 08-00

●**適用** OS：Windows 2000，Windows Server 2003

P-2462-7G84 HiRDB LDAP Option Version 8 08-02

●**適用** OS：Windows Server 2003，Windows Server 2008 R2，Windows Server 2008 Standard，Windows Server 2008 Enterprise，Windows Server 2008 (x64)，Windows XP Professional，Windows XP x64 Edition，Windows Vista Ultimate，Windows Vista Business，Windows Vista Enterprise，Windows Vista Ultimate (x64)，Windows Vista Business (x64)，Windows Vista Enterprise (x64)，Windows 7 Professional，Windows 7 Enterprise，Windows 7 Ultimate，Windows 7 Professional (x64)，Windows 7 Enterprise (x64)，Windows 7 Ultimate (x64)

P-2962-7184 HiRDB/Single Server Version 8(64) 08-51

P-2962-7384 HiRDB/Parallel Server Version 8(64) 08-51

P-2462-7P84 HiRDB Accelerator Version 8 08-03

●**適用** OS：Windows XP x64 Edition，Windows Server 2003 x64 Editions，Windows Vista (x64)，Windows Server 2008 (x64)，Windows 7 (x64)

P-2962-1184 HiRDB/Run Time Version 8(64) 08-51

P-2962-1284 HiRDB/Developer's Kit Version 8(64) 08-51

●**適用** OS：Windows 2000，Windows XP，Windows Server 2003，Windows Vista，Windows Server 2008，Windows 7

P-2662-1184 HiRDB/Run Time Version 8 08-51

P-2662-1284 HiRDB/Developer's Kit Version 8 08-51

これらのプログラムプロダクトのほかにもこのマニュアルをご利用になれる場合があります。詳細は「リリースノート」でご確認ください。

これらの製品は，ISO9001 および TickIT の認証を受けた品質マネジメントシステムで開発されました。

## ■輸出時の注意

本製品を輸出される場合には、外国為替及び外国貿易法の規制並びに米国輸出管理規則など外国の輸出関連法規をご確認の上、必要な手続きをお取りください。

なお、不明な場合は、弊社担当営業にお問い合わせください。

## ■商標類

ActiveX は，米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。

AIX は，米国およびその他の国における International Business Machines Corporation の商標です。

AMD は，Advanced Micro Devices, Inc. の商標です。

BEA WebLogic Server は，BEA Systems, Inc. の登録商標です。

CORBA は，Object Management Group が提唱する分散処理環境アーキテクチャの名称です。

DataStage，MetaBroker，MetaStage および QualityStage は，IBM Corporation の商標です。

DB2 は，米国およびその他の国における International Business Machines Corporation の商標です。

DNCWARE，ClusterPerfect は，東芝ソリューション株式会社の商標です。

gzip は，米国 FSF(Free Software Foundation) が配布しているソフトウェアです。

HACMP/6000 は，米国およびその他の国における International Business Machines Corporation の商標です。

HP C/HP-UX は，Hewlett-Packard Development Company, L.P. の商品名称です。

IBM は，米国およびその他の国における International Business Machines Corporation の商標です。

Intel は，アメリカ合衆国およびその他の国における Intel Corporation の商標です。

Itanium は，アメリカ合衆国およびその他の国における Intel Corporation の商標です。
JBuilder は，Embarcadero Technologies, Inc. の米国およびその他の国における商標です。
Linux は，Linus Torvalds 氏の日本およびその他の国における登録商標または商標です。
Microsoft および Visual Studio は，米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。
Microsoft Access は，米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。
Microsoft Office および Excel は，米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。
Motif は，Open Software Foundation,Inc. の商標です。
MS-DOS は，米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。
NLS は，Hewlett-Packard Development Company, L.P. の商品名称です。
ODBC は，米国 Microsoft Corporation が提唱するデータベースアクセス機構です。
OLE は，米国 Microsoft Corporation が開発したソフトウェア名称です。
Oracle と Java は，Oracle Corporation 及びその子会社，関連会社の米国及びその他の国における登録商標です。
OS/390 は，米国およびその他の国における International Business Machines Corporation の商標です。
PA-RISC は，Hewlett-Packard Development Company, L.P. の商標です。
POSIX は，the Institute of Electrical and Electronics Engineers, Inc. (IEEE) で制定された標準仕様です。
PowerBuilder は，Sybase,Inc. の登録商標です。
Red Hat は，米国およびその他の国で Red Hat, Inc. の登録商標もしくは商標です。
RISC System/6000 は，米国およびその他の国における International Business Machines Corporation の商標です。
TurboLinux の名称及びロゴは，TurboLinux, Inc. の商標です。
TUXEDO は，米国 UNIX System Laboratories, Inc. の商品名称です。
UNIFY2000 は，米国 Unify Corp. の商品名称です。
UNIX は，The Open Group の米国ならびに他の国における登録商標です。
VERITAS および VERITAS ロゴは，Symantec Corporation の米国およびその他の国における商標または登録商標です。
VERITAS NetBackup BusinesServer は，Symantec Corporation の米国およびその他の国における商品名称です。
Visual Basic は，米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。
Visual C++ は，米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。
Windows は，米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。
Windows NT は，米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。
Windows Server は，米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。
Windows Vista は，米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。
X/Open は，The Open Group の英国ならびに他の国における登録商標です。
X Window System は，米国 X Consortium,Inc. が開発したソフトウェアです。
イーサネットは，富士ゼロックス ( 株 ) の商品名称です。
その他記載の会社名，製品名は，それぞれの会社の商標もしくは登録商標です。

## ■発行

2013 年 10 月　3020-6-352-42

## ■著作権

All Rights Reserved. Copyright (C) 2006, 2013, Hitachi, Ltd.

## 変更内容

**変更内容** (3020-6-352-42)

<table>
<tr><th>追加・変更内容</th><th>変更個所</th></tr>
<tr><td>(2) HiRDB が作成するディレクトリ及びファイル<br>表 1-4　HiRDB が作成するディレクトリ及びファイル構成<br>［追加］<br>%PDDIR%¥spool¥tmp ｜作業用一時ファイル格納ディレクトリ</td><td>1.2.1</td></tr>
<tr><td>(4) トラブルシュート情報<br>［訂正前］<br>%PDDIR%¥spool¥shmdump¥*<br>［訂正後］<br>%PDDIR%¥spool¥pdshmdump¥*</td><td>1.2.2</td></tr>
<tr><td>(6) 運用コマンド作業用一時ファイル<br>［追加］<br>
<table>
<tr><th>項番</th><th>ファイル名又はディレクトリ名</th><th>説明</th><th>ファイルサイズの概算値</th><th>ファイル数</th><th>最大サイズの制限可否</th><th>出力される製品種別</th><th>サポートバージョン</th></tr>
<tr><td colspan="8">（略）</td></tr>
<tr><td>6</td><td>%PDDIR%¥spool¥tmp¥*</td><td>運用コマンド一時ファイルです。運用コマンド実行時に生成されます。<br>運用コマンド実行終了時に自動的に削除されます。</td><td>最大1024バイト</td><td>コマンドのプロセスIDごとに1ファイル</td><td>×</td><td>S，P</td><td>08-51-16<br>または<br>08-51-/P</td></tr>
</table>
</td><td>1.2.2</td></tr>
<tr><td>(2) 簡易セットアップツールを実行するマシン環境<br>表 3-1　簡易セットアップツールを実行するマシンの前提 OS と，Windows 版 HiRDB のセットアップ可否<br>［追加］<br>プラットフォーム：Windows<br>Windows 7 Professional　｜○<br>Windows 7 Ultimate，又は Windows 7 Enterprise　｜○<br>プラットフォーム：Windows (x64)<br>Windows 7 Professional　｜○<br>Windows 7 Ultimate，又は Windows 7 Enterprise　｜○</td><td>3.1.4</td></tr>
<tr><td>(3) Windows の設定について<br>［追加］<br>Windows のフォントサイズが規定のサイズ (96dpi) であることを確認してください。</td><td>3.1.5</td></tr>
<tr><td>表 6-2　順序番号の取得方式ごとの特徴<br>•「バッファ単位取得方式」欄<br>［訂正前］<br>発生しません。<br>［訂正後］<br>発生しません。※2<br>•「検討項目」欄<br>［訂正前］<br>順序数生成子への採番要求時の通信オーバヘッド※<br>［訂正後］<br>順序数生成子への採番要求時の通信オーバヘッド※1</td><td>6.1.7</td></tr>
</table>

<table>
<tr><th>追加・変更内容</th><th>変更個所</th></tr>
<tr><td>表 6-2 の下の注意書き<br>［訂正前］<br>注※<br>HiRDB/ パラレルサーバの場合，（中略）通信が発生します。<br>［訂正後］<br>注※ 1<br>HiRDB/ パラレルサーバの場合，（中略）通信が発生します。<br>注※ 2<br>順序番号の格納方式が列データ全置換以外の場合は，欠番が発生する可能性があります。</td><td>6.1.7</td></tr>
<tr><td>(4) インデクスの使用条件<br>表 6-4　XML 型全文検索用インデクスの使用条件<br>「XQuery 中の演算子又は関数」<br>［訂正前］<br>fn:contains<br>fn:starts-wtih<br>fn:ends-with<br>=<br>［訂正後］<br>=<br>fn:contains<br>fn:starts-wtih<br>fn:ends-with<br><br>「XML 型全文検索用インデクスの使用条件」<br>［訂正前］<br>(d)1，2，4，5，6，7<br>［訂正後］<br>(d)1，2，4，5，6，7，8</td><td>6.4.2</td></tr>
<tr><td>(4) インデクスの使用条件<br>［追加］<br>部分構造インデクス，及び XML 全文検索用インデクスの両方が使用可能な XQuery 中の演算子，又は関数を使用して検索する場合，使用されるインデクスは演算子，又は関数によって決まります。異なるインデクスを使用したい場合は，マニュアル「HiRDB Version 8 SQL リファレンス」の「使用インデクスの SQL 最適化指定」で使用するインデクスを指定することができます。<br>表 6-5　複数のインデクスが使用可能な演算子又は関数の場合に使用されるインデクス<br><table><tr><th>部分構造インデクス，及びXML型全文検索用インデクスの両方が使用可能なXQuery中の演算子又は関数</th><th>検索に使用されるインデクス</th></tr><tr><td>=</td><td>部分構造インデクス</td></tr><tr><td>fn:contains</td><td>XML型全文検索用インデクス</td></tr><tr><td>fn:starts-wtih</td><td>部分構造インデクス</td></tr><tr><td>fn:ends-with</td><td>XML型全文検索用インデクス</td></tr></table></td><td>6.4.2</td></tr>
<tr><td>(4) インデクスの使用条件<br>(b) 部分構造インデクスの使用条件<br>［訂正前］<br>なお，1.，2.，4. ～ 10. では，次のように定義したインデクスを例文中で使用します。<br>　create index idx1 on t1(c1) key using unique tag from '/root/elm1/@attr1' as varchar(10)<br>3. では，次のように定義したインデクスを例文中で使用します。<br>　create index idx2 on t1(c1) key from '/root/elm1/@attr1' as varchar(10)<br>［訂正後］<br>なお，1.，2.，4. ～ 9. では，次のように定義したインデクスを例文中で使用します。<br>　create index idx1 on t1(c1) key using unique tag from '/root/elm1/@attr1' as varchar(10)<br>　create index idx4 on t1(c1) key using unique tag from '/root/elm1/elm2<br>' as varchar(10)<br>3.，10. では，次のように定義したインデクスを例文中で使用します。<br>　create index idx2 on t1(c1) key from '/root/elm1/@attr1' as varchar(10)<br>　create index idx5 on t1(c1) key from '/root/elm1/elm2' as varchar(10)</td><td>6.4.2</td></tr>
</table>

| 追加・変更内容 | 変更個所 |
| --- | --- |
| (4) インデクスの使用条件<br>(b) 部分構造インデクスの使用条件 9.<br>［訂正前］<br>修飾名 :: = 〔接頭辞 :〕局所名<br><br>(例)<br>select c2 from t1<br>  where xmlexists('/root/child::elm1[@attr1 eq "ABC"]'<br>  passing by value c1)<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。<br>［訂正後］<br>修飾名 :: = 〔接頭辞 :〕局所名<br><br>以下，(例 1) ～ (例 3) は値比較，又は汎用比較を使用した例です。下記の例で，値比較 eq に他の値比較又は汎用比較を指定した場合も，インデクスが使用されます。<br>USING UNIQUE TAG の指定有無に関連するインデクスの使用有無は 2. と 3. を参照してください。<br>(例 1) ステップ式に @，又は attribute:: と，修飾名を指定した場合 (@ と attribute:: は同義)<br>select c2 from t1<br>  where xmlexists('/root/child::elm1[@attr1 eq "ABC"]'<br>  passing by value c1)<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。<br>(例 2) ステップ式に child:: を指定，又は child:: を省略し，修飾名を指定した場合<br>select c2 from t1<br>  where xmlexists('/root/child::elm1[child::elm2 eq "ABC"]'<br>  passing by value c1)<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。<br>(例 3) ステップ式に文脈項目式を指定した場合 (値比較は 2. に該当する場合だけ)<br>select c2 from t1<br>  where xmlexists('/root/child::elm1/elm2[ . eq "ABC"]'<br>  passing by value c1)<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。<br>以下，(例 4) ～ (例 6) は XQuery 関数を使用した例です。下記の例で，fn:starts-with に他の XQuery 関数を指定した場合も，インデクスが使用されます。<br>(例 4) ステップ式に @，又は attribute:: と，修飾名を指定した場合 (@ と attribute:: は同義)<br>select c2 from t1<br>  where xmlexists('/root/elm1[fn:starts-with(@attr1 ,"ABC")]'<br>  passing by value c1)<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。<br>(例 5) ステップ式に child:: を指定，又は child:: を省略し，修飾名を指定した場合<br>select c2 from t1<br>  where xmlexists('/root/elm1[fn:starts-with(elm2 ,"ABC")]'<br>  passing by value c1)<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。<br>(例 6) ステップ式に文脈項目式を指定した場合<br>select c2 from t1<br>  where xmlexists('/root/elm1/@attr1[fn:starts-with( . ,"ABC")]'<br>  passing by value c1)<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。 | 6.4.2 |

| 追加・変更内容 | 変更個所 |
|---|---|
| (4) インデクスの使用条件<br>(b) 部分構造インデクスの使用条件 10.<br>［訂正前］<br>修飾名 :: = 〔接頭辞 :〕局所名<br><br>(例)<br>`select c2 from t1`<br>`  where xmlexists('/root/elm1[starts-with(@attr1 ,"ABC")]'`<br>`  passing by value c1)`<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。<br>［訂正後］<br>修飾名 :: = 〔接頭辞 :〕局所名<br><br>下記の例で，fn:starts-with に他の XQuery 関数を指定した場合も，インデクスを使用します。<br>(例 1) ステップ式終端に @，又は attribute:: と，修飾名を指定した場合（@ と attribute:: は同義）<br>`select c2 from t1`<br>`  where xmlexists('/root/elm1[fn:starts-with(@attr1 ,"ABC")]'`<br>`  passing by value c1)`<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。<br>(例 2) ステップ式終端に文脈項目式を指定した場合<br>`select c2 from t1`<br>`  where xmlexists('/root/elm1/@attr1[fn:starts-with( . ,"ABC")]'`<br>`  passing by value c1)`<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。 | 6.4.2 |
| (4) インデクスの使用条件<br>(d) XML 型全文検索用インデクスの使用条件 2.<br>［訂正前］<br>注※　HiRDB XML Extension のバージョンが 08-04 以降の場合に指定できます。<br>(例)<br>`select c2 from t2`<br>`  where xmlexists('/root/child::elm1[fn:contains(@attr1,"ABC")]'`<br>`  passing by value c1)`<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。<br>［訂正後］<br>下記の例で，fn:contains に他の XQuery 関数を指定した場合も，インデクスを使用します。<br>(例 1) 第一引数に @，又は attribute:: と，名前テストを指定した場合（@ と attribute:: は同義）<br>`select c2 from t2`<br>`  where xmlexists('/root/child::elm1[fn:contains(@attr1,"ABC")]'`<br>`  passing by value c1)`<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。<br>(例 2) 第一引数に @，又は attribute:: と，属性テストを指定した場合（@ と attribute:: は同義）<br>`select c2 from t2`<br>`  where xmlexists('/root/child::elm1[fn:contains(@attribute(),"ABC")]'`<br>`  passing by value c1)`<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。<br>(例 3) 第一引数に属性テストのみを指定した場合<br>`select c2 from t2`<br>`  where xmlexists('/root/child::elm1[fn:contains(attribute(),"ABC")]'`<br>`  passing by value c1)`<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。<br>(例 4) 第一引数に文脈項目式を指定した場合<br>`select c2 from t2`<br>`  where xmlexists('/root/child::elm1/text()[fn:contains( . ,"ABC")]'`<br>`  passing by value c1)`<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。 | 6.4.2 |

| 追加・変更内容 | 変更個所 |
| --- | --- |
| (4) インデクスの使用条件<br>(d) XML 型全文検索用インデクスの使用条件 3.<br>［訂正前］<br>(例)<br>select c2 from t2<br>  where xmlexists('/root/elm1[hi-fn:contains(text(),"""ABC AND DEF""")]'<br>  passing by value c1)<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。<br>［訂正後］<br>(例 1) 第一引数にテキストを指定した場合<br>select c2 from t2<br>  where xmlexists('/root/elm1[hi-fn:contains(text(),"""ABC AND DEF""")]'<br>  passing by value c1)<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。<br>select c2 from t2<br>  where xmlexists('/root/elm1[hi-fn:contains(descendant::text(),"""ABC AND<br>DEF""")]'<br>  passing by value c1)<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。<br>(例 2) 第一引数にテキストテストおよび文脈項目式を指定した場合<br>select c2 from t2<br>  where xmlexists('/root/elm1[hi-fn:contains(text()/.,"""ABC AND DEF""")]'<br>  passing by value c1)<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。<br>select c2 from t2<br>  where xmlexists('/root/elm1[hi-fn:contains(descendant::text()/.,"""ABC AND<br>DEF""")]'<br>  passing by value c1)<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。<br>(例 3) 第一引数に文脈項目式を指定した場合<br>select c2 from t2<br>  where xmlexists('/root/elm1/text()[hi-fn:contains( . ,"""ABC AND DEF""")]'<br>  passing by value c1)<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。<br>select c2 from t2<br>  where xmlexists('/root/elm1/ descendant::text()[hi-fn:contains( . ,"<br>""ABC AND DEF""")]'<br>  passing by value c1)<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。<br>(例 4) 第一引数に @ 又は attribute:: と，名前テストを指定した場合（@ と attribute:: は同義）<br>select c2 from t2<br>  where xmlexists('/root/elm1[hi-fn:contains(@attr1,"""ABC AND DEF"<br>""")]'<br>  passing by value c1)<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。<br>(例 5) 第一引数に @，又は attribute:: と，属性テストを指定した場合（@ と attribute:: は同義）<br>select c2 from t2<br>  where xmlexists('/root/elm1[hi-fn:contains(@attribute(),"""ABC AND DEF""")]'<br>  passing by value c1)<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。<br>(例 6) 第一引数に属性テストのみを指定した場合<br>select c2 from t2<br>  where xmlexists('/root/elm1[hi-fn:contains(attribute(),"""ABC AND DEF")]'<br>  passing by value c1)<br><br>注　アンダーライン部分が上記の形式と一致する部分構造パスです。 | 6.4.2 |

| 追加・変更内容 | 変更個所 |
|---|---|
| (4) インデクスの使用条件<br>(d) XML 型全文検索用インデクスの使用条件<br>［追加］<br>8. 検索対象の XML 型の列に定義した全文検索インデクスに，以下のどのプラグインオプションも指定していない。<br>プラグインオプションの詳細は，マニュアル「HiRDB XML 拡張機能 HiRDB XML Extension Version 8」の「2.5.4　インデクスの定義」を参照してください。<br>• DELcode= ファイル名<br>• NOindex= ファイル名<br>• ENGLISH<br>• ENGLISH_STANDARD<br><br>（例）<br>create index idx6 using type ixxml on t2(c1) in (LOB1) PLUGIN'SAMECASE=ON,SAMEWIDE=ON,SAMEY=ON,SAMED=ON,DELcode=ON'<br><br>（アンダーラインがインデクスを使用可能なオプションだけを指定した全文検索インデクス） | 6.4.2 |
| (4) インデクスの使用条件<br>(e) インデクスを使用しない場合<br>［訂正前］<br>次に示す場合は，部分構造インデクス又は XML 型全文検索用インデクスを使用しません。<br>［訂正後］<br>次に示す 1. ～ 4. の場合は，部分構造インデクス又は XML 型全文検索用インデクスを使用しません。<br>なお，1.，3. では，次のように定義したインデクスを例文中で使用します。<br>create index idx1 on t1(c1) key using unique tag from '/root/elm1/@attr1' as archar(10)<br><br>2.，4. では次のように定義したインデクスを例文中で使用します。<br>create index idx3 using type ixxml on t2(c1) in (LOB1) | 6.4.2 |
| (4) インデクスの使用条件<br>(e) インデクスを使用しない場合<br>［追加］<br>4. 論理演算子（OR，AND）で XMLEXISTS 述語を多数連結した場合<br>HiRDB が見積もるアクセスコストにより，インデクスを使用しない方が最適なアクセスパスとなると判断し，XMLEXISTS 述語の評価にインデクスを使用しません。hi-fn:contains 関数を指定した場合はインデクスだけで評価できないため，SQL エラーとなります。<br>（例：変更前）<br>select c2 from t2<br>  where xmlexists('/root/elm1[hi-fn:contains(text(),"""01ABC""")]' passing by value c1)<br>  or xmlexists('/root/elm1[hi-fn:contains(text(),"""02ABC""")]' passing by value c1)<br>  ・・・(省略)・・・<br>  or xmlexists('/root/elm1[hi-fn:contains(text(),"""30ABC""")]' passing by value c1)<br><br>注　アンダーライン部分が，単独の指定ではインデクスを使用する XMLEXISTS 述語を 30 個指定した条件です。<br>次のように変更すると，インデクスを使用するようになります。<br>（例：変更後）<br>select c2 from t2 with index(idx3,idx3)<br>where xmlexists('/root/elm1[hi-fn:contains(text(),"""01ABC""")]' passing by value c1)<br>or xmlexists('/root/elm1[hi-fn:contains(text(),"""02ABC""")]' passing by value c1)<br>  ・・・(省略)・・・<br>or xmlexists('/root/elm1[hi-fn:contains(text(),"""30ABC""")]' passing by value c1)<br><br>注　アンダーライン部分が，複数インデク利用の実行に必要な種類のインデクスを指定したインデクスの SQL 最適化指定です。 | 6.4.2 |

<table>
<tr><th>追加・変更内容</th><th>変更個所</th></tr>
<tr><td>(1) トランザクションマネジャが OpenTP1 の場合<br>［削除］<br>ただし，次の表で示す条件によっては，OpenTP1 でトランザクションの移行を使用する設定をしても（コミット最適化及びプリペア最適化を抑止しない設定をしても），HiRDB ではトランザクションの移行を使用しない設定ができます。</td><td>7.2.4</td></tr>
<tr><td>(1) トランザクションマネジャが OpenTP1 の場合<br>表 7-2　OpenTP1 の trnstring オペランドと HiRDB の PDXAMODE オペランドの関係<br>［訂正前］
<table>
<tr><th colspan="3">条件</th><th rowspan="2">PDXAMODE オペランドの値</th></tr>
<tr><th>trnstringオペランドの指定</th><th colspan="2">OpenTP1の状態</th></tr>
<tr><td>-dオプションを省略</td><td colspan="2">同一OpenTP1内では一つのUAPだけでグローバルトランザクションを構成している場合</td><td>0又は1</td></tr>
<tr><td rowspan="2">-dオプションを指定</td><td rowspan="2">同一OpenTP1内では複数のUAPでグローバルトランザクションを構成している場合</td><td>一つのUAPだけがHiRDB XAライブラリとリンクしている場合</td><td>0又は1</td></tr>
<tr><td>複数のUAPがHiRDB XAライブラリとリンクしている場合</td><td>1</td></tr>
<tr><td colspan="2">－</td><td>0</td></tr>
</table>
［訂正後］
<table>
<tr><th>trnstringオペランドの指定</th><th>PDXAMODEオペランドの値</th></tr>
<tr><td>-dオプションを省略</td><td>1</td></tr>
<tr><td>-dオプションを指定</td><td>0</td></tr>
</table></td><td>7.2.4</td></tr>
<tr><td>(1) トランザクションマネジャが OpenTP1 の場合<br>［削除］<br>また，OpenTP1 と HiRDB を接続したシステムでトランザクションの移行を使用するかどうかは，次に示す適用基準で判断してください。<br>適用基準<br>クライアント UAP 及びサーバ UAP が次に示す条件をすべて満たす場合は，トランザクションの移行を使用してください。<br>1. クライアント UAP 及びサーバ UAP が同一 OpenTP1 環境下にあります。<br>2. クライアント UAP 及びサーバ UAP が HiRDB XA ライブラリにリンクされています。<br>3. クライアント UAP 及びサーバ UAP が HiRDB をアクセスしています。※<br><br>注※<br>次に示すどちらかの条件を満たす場合のことです。<br>1. HiRDB をアクセスする SQL を発行している場合<br>2. OpenTP1 システム定義のトランザクションサービス定義／ユーザサービス定義の trn_rm_open_close_scope オペランドに process を指定又は省略している場合</td><td>7.2.4</td></tr>
</table>

<table>
<tr><th>追加・変更内容</th><th>変更個所</th></tr>
<tr><td>
(2) 注意事項　6.<br>
［訂正前］<br>
回復不要 FES ユニットが稼働しているのに，ネットワーク障害などでそのユニットのステータス情報が STOP(A) になった場合，障害が回復してシステムマネジャからそのユニットに通信できるようになると，システムマネジャがそのユニットを自動的に強制停止してから再度開始します。ただし，システムマネジャからそのユニットに通信できるようになる前にユニットが停止した場合はシステムマネジャは強制停止及び再度開始をしません。停止しているユニットは，必要に応じて pdstart コマンドを実行してユニットを開始してください。なお，次の場合はそれぞれ状況に応じて対策してください。<br>
• 該当するユニットに対してエラーメッセージが表示されている場合<br>
ユニットを開始する前に，メッセージからエラー要因を調査し，対策してください。ユニットが停止している場合，syslogfile システムログに KFPS01841-I 又は KFPS01821-E メッセージが表示されます。pdstart コマンド実行時，KFPS01853-W，KFPS01863-E，及び KFPS05619-E メッセージは表示されません。<br>
• システムマネジャがユニットを再度開始する場合，起動処理に失敗してユニットが停止状態のままのとき<br>
メッセージから起動処理に失敗した原因を調査し，対策してください。その後，必要に応じて該当するユニットを pdstart コマンドで開始してください。<br>
<br>
［訂正後］<br>
回復不要 FES ユニットが稼働しているのに，ネットワーク障害などで回復不要 FES ユニットのステータス情報が STOP(A) になった場合，障害が回復してシステムマネジャを配置するユニットから回復不要 FES ユニットに通信できるようになると，システムマネジャが回復不要 FES ユニットを自動的に強制停止してから再度開始します。自動的に強制停止してから再度開始する契機を次に示します。<br>
• システムマネジャを配置するユニットのユニット監視プロセスが，STOP(A) になっている回復不要 FES ユニットの稼動状態を確認し，KFPS05288-I メッセージを出力したとき。<br>
• システムマネジャを配置するユニットが再度開始する際に，システムマネジャが，STOP(A) になっている回復不要 FES ユニットの稼動状態を確認したとき。<br>
<br>
(1) 回復不要 FES ユニットの起動処理が正常に終了していない場合の対処<br>
(a) システムマネジャを配置するユニット，及びディクショナリサーバを配置するユニットの稼働状態を pdls -d ust コマンドで確認します。<br>
(b) これらのユニットが稼働していない場合，pdstart -q コマンドで開始します。<br>
(c) これらのユニットが稼働している場合，又はこれらユニットを開始しても KFPS05110-I メッセージを出力しない場合は，(2) の対処をします。<br>
(2) KFPS05110-I メッセージを出力しない場合の対処<br>
回復不要 FES ユニットの稼働状態を pdls -d ust コマンドで確認し，実行結果に応じて，次に示す対処をします。<br>
<table>
<tr><th>回復不要FESユニットの稼働状態<br>（pdls -d ustコマンドの実行結果）</th><th>対処方法</th></tr>
<tr><td>STOP（停止状態）</td><td>pdstart -qコマンドを実行して，回復不要FESユニットを再開始します。</td></tr>
<tr><td>PAUSE（プロセスサーバプロセスの再起動中断状態）</td><td>1. KFPS00715-Eメッセージ，及びそれ以前にイベントログに出力されたメッセージを参照して，障害要因を取り除いた後，HiRDBのサービスを再起動します。<br>2. pdstart -qコマンドを実行して，回復不要FESユニットを再開始します。</td></tr>
<tr><td>STARTING（開始途中）</td><td rowspan="3">1. pdstop -zコマンドを実行して，回復不要FESユニットを強制終了します。<br>2. pdstart -qコマンドを実行して，回復不要FESユニットを再開始します。</td></tr>
<tr><td>ONLINE（稼働状態）</td></tr>
<tr><td>STOPPING（停止途中）</td></tr>
</table>
なお，システムマネジャから回復不要 FES ユニットに通信できるようになる前にユニットが停止した場合は，システムマネジャは強制停止及び再度開始をしません。
</td><td>9.1.4</td></tr>
</table>

| 追加・変更内容 | 変更個所 |
|---|---|
| (2) 注意事項　7.<br>［訂正前］<br>回復不要 FES から分岐してバックエンドサーバやディクショナリサーバで実行しているトランザクションは，コミット決着時に分岐先のバックエンドサーバ及びディクショナリサーバ間で決着の同期合わせを行います。このため，同期合わせのときに，ほかのバックエンドサーバやディクショナリサーバがトランザクション処理を実行できない状態（系切り替え中，サーバ停止状態，サーバ開始準備中，又はサーバ停止準備中）になっていると，トランザクション第 1 状態が READY 又は COMMIT で待ち合わせを行うことがあります。この場合，トランザクション処理を実行できない状態になっている原因を該当するサーバで対策して，トランザクションの決着処理が続行できるようにしてください。<br>［訂正後］<br>回復不要 FES から分岐して，ほかのサーバで実行しているトランザクションは，コミット決着時に分岐先のサーバ間で決着の同期合わせを行います。このため，同期合わせのときに，分岐先サーバのいずれかがトランザクション処理を実行できない状態（系切り替え中，サーバ停止状態，サーバ開始準備中，又はサーバ停止準備中）になっていると，トランザクション第 1 状態が READY 又は COMMIT で待ち合わせを行うことがあります。この場合，トランザクション処理を実行できない状態になっている原因を該当するサーバで対策して，トランザクションの決着処理が続行できるようにしてください。 | 9.1.4 |
| (3) ビュー表の作成方法<br>［訂正前］<br>• 最大 15 個の実表又は外部表を一つにしたビュー表<br>［訂正後］<br>• 最大 64 個の実表を基にした一つのビュー表 | 12.6 |
| 注意事項<br>［訂正前］<br>• 可変長の行（BINARY 型を含む）に空き領域の再利用機能が使用された場合，追加と削除が同量の場合でも，セグメント数が増加する場合があります。<br>［訂正後］<br>• 非 FIX 表の場合，空き領域再利用機能を適用していても，次のような更新を行うとデータの追加と削除が同量でもセグメント数が増加する場合があります。<br>・↓（ページ長− 48）÷（行長＋ 2）↓件のデータを挿入し，行長が短くなるデータ（NULL 値含む），又は別のページに分岐するようなデータに更新した後削除する操作を繰り返す。<br>そのため，定期的にデータベース再編成ユティリティ（pdrorg）によるデータの再編成を実施してください。 | 14.5.7 |
| 表 15-1　HiRDB/ シングルサーバの共用メモリの詳細<br>［訂正前］<br>作成契機　｜ユニット起動時（高速系切り替え機能使用時の待機ユニットの起動を含む）<br>削除契機　｜次回ユニット起動時（高速系切り替え機能使用時の待機ユニットの起動を含む）<br>［訂正後］<br>作成契機　｜ユニット起動時（ユーザサーバホットスタンバイまたは高速系切り替え機能使用時の待機ユニットの起動を含む）<br>削除契機　｜ユニット起動時（ユーザサーバホットスタンバイまたは高速系切り替え機能使用時の待機ユニットの起動を含む） | 15.1.1 |
| 表 15-2　HiRDB/ シングルサーバが使用するメモリ所要量<br>項目「ユニットコントローラ全プロセスが使用するプロセス固有領域」<br>メモリ所要量（単位：キロバイト）欄<br>［訂正前］<br>**●インメモリデータ処理を行う場合，加算します。**<br>＋ K ×（pd_max_users の値× 2 ＋ 7）<br>［訂正後］<br>**●インメモリデータ処理を行う場合，加算します。**<br>＋↑ {K ×（pd_max_users の値× 2 + 7)} ÷ 1024 ↑<br>**●通信トレース格納最大数を変更する場合，加算します。**<br>+ ↑ M ÷ 1024 ↑ | 15.1.2 |

| 追加・変更内容 | 変更個所 |
|---|---|
| 表 15-2　HiRDB/ シングルサーバが使用するメモリ所要量<br>項目「シングルサーバプロセスが使用するプロセス固有領域」<br>メモリ所要量（単位：キロバイト）欄<br>• pd_work_buff_mode=each 指定時<br><br>［訂正前］<br>**●インメモリデータ処理を行う場合，加算します。**<br>＋ K ×（pd_max_users の値＋ 3）<br>［訂正後］<br>**●インメモリデータ処理を行う場合，加算します。**<br>＋↑｛K ×（pd_max_users の値＋ 3)｝÷ 1024 ↑<br>**●通信トレース格納最大数を変更する場合，加算します。**<br>＋↑ P ÷ 1024 ↑<br>• pd_work_buff_mode=pool 指定又は省略時<br><br>［訂正前］<br>**●インメモリデータ処理を行う場合，加算します。**<br>＋ K<br>［訂正後］<br>**●インメモリデータ処理を行う場合，加算します。**<br>＋↑｛K ×（pd_max_users の値＋ 3)｝÷ 1024 ↑<br>**●通信トレース格納最大数を変更する場合，加算します。**<br>＋↑ P ÷ 1024 ↑ | 15.1.2 |
| 表 15-2 の下の注意書き<br>［訂正前］<br>K：pd_max_resident_rdarea_no オペランドに 1 以上を指定している場合に，次の値を加算します。<br>　1648 ＋ 16 × pd_max_resident_rddarea_no の値＋ 16 × pd_max_resident_rddarea_shm_no の値<br>L：（略）<br>［訂正後］<br>K：pd_max_resident_rdarea_no オペランドに 1 以上を指定している場合に，次の値を加算します。<br>　1648 ＋ 16 × pd_max_resident_rdarea_no の値＋ 16 × pd_max_resident_rdarea_shm_no の値<br>L：（略）<br>M：通信トレース処理で必要なメモリ所要量<br>　32 ビットモードの場合：<br>　（16 ×（N － 1024）× 2）×（pd_max_server_process 値－ pd_max_users 値－ 3）<br>　64 ビットモードの場合：<br>　（32 ×（N － 1024）× 2）×（pd_max_server_process 値－ pd_max_users 値－ 3）<br>N：ユニット制御情報定義として有効な pd_pth_trace_max 値です。オペランド指定値を 2 のべき乗の値に切り上げた値になります。<br>P：通信トレース処理で必要なメモリ所要量<br>　32 ビットモードの場合：<br>　（16 ×（Q － 1024）× 2）×（pd_max_users 値＋ 3）<br>　64 ビットモードの場合：<br>　（32 ×（Q － 1024）× 2）×（pd_max_users 値＋ 3）<br>Q：シングルサーバ定義として有効な pd_pth_trace_max 値です。オペランド指定値を 2 のべき乗の値に切り上げた値になります。 | 15.1.2 |
| (1) 32 ビットモードの HiRDB の場合<br>プロセスの種類：ユニット共通<br>［訂正前］<br>ａ：25328 ＋ p × 4<br>［訂正後］<br>ａ：25360 ＋ p × 4 | 15.1.3 |
| (2) 64 ビットモードの HiRDB の場合<br>プロセスの種類：ユニット共通<br>［訂正前］<br>ａ：34736 ＋ ｐ × 4<br>［訂正後］<br>ａ：34768 ＋ ｐ × 4 | 15.1.3 |

| 追加・変更内容 | 変更個所 |
|---|---|
| (1) グローバルバッファが使用する共用メモリを実メモリ上に固定しない場合<br>表　計算式 1<br>［訂正前］<br>**● 32 ビットモードの場合**<br>$\sum_{i=1}^{n}$ {<br>↑ {736＋64＋（280＋64※1）×（Pi＋4）<br>＋（124＋80※2＋96×A×Mi）×Ui} ÷4096↑×4096<br>＋Si×｛Pi＋4＋（Ui×Mi×A）｝<br>｝÷1024<br>**● 64 ビットモードの場合**<br>$\sum_{i=1}^{n}$ {<br>↑ {928＋64＋（464＋112※1）×（Pi＋4）<br>＋（176＋96※2＋136×A×Mi）×Ui} ÷4096↑×4096<br>＋Si×｛Pi＋4＋（Ui×Mi×A）｝<br>｝÷1024<br>［訂正後］<br>**● 32 ビットモードの場合**<br>$\sum_{i=1}^{n}$ {《管理領域部》＋《データ格納部》｝÷1024<br>《管理領域部》<br>↑ {736 ＋ 64 ＋（296 ＋ 64※1）×（Pi ＋ 4）<br>＋（124 ＋ 80※2＋ 96 × A × Mi）× Ui} ÷ 4096 ↑× 4096<br>《データ格納部》<br>Si ×｛Pi ＋ 4 ＋（Ui × Mi × A)｝<br>**● 64 ビットモードの場合**<br>$\sum_{i=1}^{n}$ {《管理領域部》＋《データ格納部》｝÷1024<br>《管理領域部》<br>↑ {928 ＋ 64 ＋（480 ＋ 112※1）×（Pi ＋ 4）<br>＋（176 ＋ 96※2＋ 136 × A × Mi）× Ui} ÷ 4096 ↑× 4096<br>《データ格納部》<br>Si ×｛Pi ＋ 4 ＋（Ui × Mi × A)｝ | 15.1.5 |
| (1) グローバルバッファが使用する共用メモリを実メモリ上に固定しない場合<br>表　計算式 2<br>［訂正前］<br>**● 32 ビットモードの場合**<br>{↑［((s × 1024 ÷ 4) ÷ 8) ＋ 112］÷ 4096 ↑× 4096 ×↑ a ÷（s × 1024）↑<br>｝÷ 1024<br>**● 64 ビットモードの場合**<br>{↑［((s × 1024 ÷ 4) ÷ 8) ＋ 144］÷ 4096 ↑× 4096 ×↑ a ÷（s × 1024）↑<br>｝÷1024<br>［訂正後］<br>**● 32 ビットモードの場合**<br>{↑［((s × 1024 ÷ 4) ÷ 8) ＋ 112］÷ 4096 ↑× 4096 ×↑ a ÷（s × 1024）↑<br>｝<br>**● 64 ビットモードの場合**<br>{↑［((s × 1024 ÷ 4) ÷ 8) ＋ 144］÷ 4096 ↑× 4096 ×↑ a ÷（s × 1024）↑<br>｝ | 15.1.5 |

| 追加・変更内容 | 変更個所 |
| --- | --- |
| (2) グローバルバッファが使用する共用メモリを実メモリ上に固定する場合<br>表　計算式 1<br>［訂正前］<br>**● 32 ビットモードの場合**<br>n<br>Σ {<br>i=1<br>↑ {↑ {736＋64＋（280＋64※1）×（Pi＋4）<br>＋（124＋80※2＋96×A×Mi）×Ui} ÷4096↑×4096<br>＋Si×｛Pi＋4＋（Ui×Mi×A）｝｝÷p↑×p<br>｝÷1024<br>**● 64 ビットモードの場合**<br>n<br>Σ {<br>i=1<br>↑ {↑ {928＋64＋（464＋112※1）×（Pi＋4）<br>＋（176＋96※2＋136×A×Mi）×Ui} ÷4096↑×4096<br>＋Si×｛Pi＋4＋（Ui×Mi×A）｝｝÷p↑×p<br>｝÷1024<br>［訂正後］<br>**● 32 ビットモードの場合**<br>n<br>Σ {↑ {《管理領域部》+《データ格納部》} ÷ p ↑× p} ÷ 1024<br>i=1<br>《管理領域部》:<br>↑ {↑ {736 ＋ 64 ＋（296 ＋ 64※1）×（Pi ＋ 4）<br>＋（124 ＋ 80※2＋ 96 × A × Mi）× Ui} ÷ 4096 ↑× 4096<br>《データ格納部》<br>Si ×｛Pi ＋ 4 ＋（Ui × Mi × A)｝<br>**● 64 ビットモードの場合**<br>n<br>Σ {↑ {《管理領域部》＋《データ格納部》} ÷ p ↑× p} ÷ 1024<br>i=1<br>《管理領域部》:<br>↑ {↑ {928 ＋ 64 ＋（480 ＋ 112※1）×（Pi ＋ 4）<br>＋（176 ＋ 96※2＋ 136 × A × Mi）× Ui} ÷ 4096 ↑× 4096<br>《データ格納部》:<br>Si ×｛Pi ＋ 4 ＋（Ui × Mi × A)｝ | 15.1.5 |
| (2) グローバルバッファが使用する共用メモリを実メモリ上に固定する場合<br>表　計算式 2<br>［訂正前］<br>**● 32 ビットモードの場合**<br>{↑ {↑［((s × 1024 ÷ 4) ÷ 8) ＋ 112］÷ 4096 ↑× 4096 ×↑ a ÷（s × 1024）↑} ÷ p ↑× p} ÷ 1024<br>**● 64 ビットモードの場合**<br>{↑ {↑［((s × 1024 ÷ 4) ÷ 8) ＋ 144］÷ 4096 ↑× 4096 ×↑ a ÷（s × 1024）↑} ÷ p ↑× p} ÷ 1024<br>［訂正後］<br>**● 32 ビットモードの場合**<br>{↑ {↑［((s × 1024 ÷ 4) ÷ 8) ＋ 112］÷ 4096 ↑× 4096 ×↑ a ÷（s × 1024）↑} ÷ p ↑× p}<br>**● 64 ビットモードの場合**<br>{↑ {↑［((s × 1024 ÷ 4) ÷ 8) ＋ 144］÷ 4096 ↑× 4096 ×↑ a ÷（s × 1024）↑} ÷ p ↑× p} | 15.1.5 |

| 追加・変更内容 | 変更個所 |
|---|---|
| 表 15-6　HiRDB/ パラレルサーバの共用メモリの詳細<br>［訂正前］<br>作成契機　｜ユニット起動時（高速系切り替え機能使用時の待機ユニットの起動を含む）<br>削除契機　｜次回ユニット起動時（高速系切り替え機能使用時の待機ユニットの起動を含む）<br>［訂正後］<br>作成契機　｜ユニット起動時（ユーザサーバホットスタンバイまたは高速系切り替え機能使用時の待機ユニットの起動を含む）<br>削除契機　｜ユニット起動時（ユーザサーバホットスタンバイまたは高速系切り替え機能使用時の待機ユニットの起動を含む） | 15.2.1 |
| 表 15-7　HiRDB/ パラレルサーバの各ユニットが使用するメモリ所要量<br>項目「ユニットコントローラ全プロセスが使用するプロセス固有領域」<br>メモリ所要量（単位：キロバイト）欄<br>［訂正前］<br>**●インメモリデータ処理を行う場合，加算します。**<br>＋ T ×（pd_max_bes_process の値× 2 ＋ 7）×ユニット内 BES 数<br>［訂正後］<br>**●インメモリデータ処理を行う場合，加算します。**<br>＋↑｛T ×（pd_max_bes_process の値× 2 ＋ 7）×ユニット内 BES 数｝÷ 1024 ↑<br>**●通信トレース格納最大数を変更する場合，加算します。**<br>＋↑ V ÷ 1024 ↑ | 15.2.2 |
| 表 15-7　HiRDB/ パラレルサーバの各ユニットが使用するメモリ所要量<br>項目「フロントエンドサーバ」<br>メモリ所要量（単位：キロバイト）欄<br>［訂正前］<br>（N ＋ h ＋ m ＋ p ＋ q）×（b ＋ 3）＋ 100 ＋ y<br>［訂正後］<br>（N + h + m + p + q）×（b + 3）＋ 100 ＋ y<br>**●通信トレース格納最大数を変更する場合，加算します。**<br>+ ↑ W ÷ 1024 ↑ | 15.2.2 |
| 表 15-7　HiRDB/ パラレルサーバの各ユニットが使用するメモリ所要量<br>項目「ディクショナリサーバ」<br>• pd_work_buff_mode=each 指定時<br><br>［訂正前］<br>｛(P ＋ i ＋ m ＋ r ＋ t）×（b ＋ 3)｝＋（a ＋ 9）× 2 ＋ 100 ＋ y ＋ S<br>［訂正後］<br>｛(P ＋ i ＋ m ＋ r ＋ t）×（b ＋ 3)｝＋（a ＋ 9）× 2 ＋ 100 ＋ y ＋ S<br>**●通信トレース格納最大数を変更する場合，加算します。**<br>＋↑ W ÷ 1024 ↑<br><br>• pd_work_buff_mode=pool 指定又は省略時<br><br>［訂正前］<br>｛(P ＋ i ＋ m ＋ r ＋ t）×（b ＋ 3)｝＋ a ＋ 9 ＋↑ a ÷ 128 × 0.1 ↑＋ 100 ＋ n ＋ y ＋ S<br>［訂正後］<br>｛(P ＋ i ＋ m ＋ r ＋ t）×（b ＋ 3)｝＋ a ＋ 9 ＋↑ a ÷ 128 × 0.1 ↑＋ 100 ＋ n ＋ y ＋ S<br>**●通信トレース格納最大数を変更する場合，加算します。**<br>＋↑ W ÷ 1024 ↑ | 15.2.2 |

| 追加・変更内容 | 変更個所 |
|---|---|
| 表 15-7　HiRDB/ パラレルサーバの各ユニットが使用するメモリ所要量<br>項目「バックエンドサーバ」<br>• pd_work_buff_mode=each 指定時<br><br>［訂正前］<br>{Q ＋ g ＋（a ＋ 9）× c ＋ i ＋ m ＋ r ＋ t} ×（b ＋ 3）＋ 100 ＋ y ＋ S<br>**●インメモリデータ処理を行う場合，加算します。**<br>＋ T ×（pd_max_users の値＋ 3）<br>［訂正後］<br>{Q ＋ g ＋（a ＋ 9）× c ＋ i ＋ m ＋ r ＋ t} ×（b ＋ 3）＋ 100 ＋ y ＋ S<br>**●インメモリデータ処理を行う場合，加算します。**<br>＋↑ {T ×（pd_max_users の値＋ 3)} ÷ 1024 ↑<br>**●通信トレース格納最大数を変更する場合，加算します。**<br>＋↑ W ÷ 1024 ↑<br><br>• pd_work_buff_mode=pool 指定又は省略時<br><br>［訂正前］<br>●インメモリデータ処理を行う場合，加算します。<br>＋ T ×（pd_max_users の値＋ 3）<br>［訂正後］<br>＋↑ {T × (pd_max_users の値＋ 3)} ÷ 1024 ↑<br>**●通信トレース格納最大数を変更する場合，加算します。**<br>＋↑ W ÷ 1024 ↑ | 15.2.2 |
| 表 15-7 の下の注意書き<br>［追加］<br>x：ユニット制御情報定義として有効な pd_pth_trace_max 値です。オペランド指定値を 2 のべき乗の値に切り上げた値になります。<br>V：通信トレース処理で必要なメモリ所要量<br>32 ビットモードの場合：<br>　（16 ×（x － 1024）× 2）×（pd_max_server_process 値－ w）<br>64 ビットモードの場合：<br>　（32 ×（x － 1024）× 2）×（pd_max_server_process 値－ w）<br>W：通信トレース処理で必要なメモリ所要量<br>ユニット内の各サーバプロセスに関して算出した下記の値<br>32 ビットモードの場合：<br>（16 ×（Z － 1024）× 2）×（最大起動プロセス数＋ 3）<br>64 ビットモードの場合：<br>（32 ×（Z － 1024）× 2）×（最大起動プロセス数＋ 3）<br>最大起動プロセス数については，マニュアル「HiRDB Version 8 システム定義」を参照してください。<br>Z：各サーバ定義として有効な pd_pth_trace_max 値です。<br>　オペランド指定値を 2 のべき乗の値に切り上げた値になります。 | 15.2.2 |
| (1) 32 ビットモードの場合<br>プロセスの種類：ユニット共通<br>［訂正前］<br>ａ：26720 ＋ v × 4 × 34<br>［訂正後］<br>ａ：26752 ＋ v × 4 × 34 | 15.2.3 |
| (2) 64 ビットモードの場合<br>プロセスの種類：ユニット共通<br>［訂正前］<br>ａ：36128 ＋ v × 4 × 34<br>［訂正後］<br>ａ：36160 ＋ v × 4 × 34 | 15.2.3 |

| 追加・変更内容 | 変更個所 |
|---|---|
| (1) フロントエンドサーバが使用する共用メモリの計算式<br>**● 32 ビットモードの場合**<br>［訂正前］<br>＋ 32 ＋ {16 ×（f ＋ 1）× g} ＋ 20000 ＋ 112 × B<br>［訂正後］<br>＋ 36 ＋ {28 ×（f ＋ 1）× g} ＋ 20000 ＋ 112 × B | 15.2.4 |
| (1) フロントエンドサーバが使用する共用メモリの計算式<br>**● 64 ビットモードの場合**<br>［訂正前］<br>＋ 48 ＋ {16 ×（f ＋ 1）× g} ＋ 20000 ＋ 112 × B<br>［訂正後］<br>＋ 56 ＋ {40 ×（f ＋ 1）× g} ＋ 20000 ＋ 112 × B | 15.2.4 |
| (1)(a) グローバルバッファが使用する共用メモリを実メモリ上に固定しない場合<br>表　計算式 1<br>［訂正前］<br>**● 32 ビットモードの場合**<br>$\sum_{i=1}^{n}$ {<br>↑ {736 ＋ 64 ＋（280 ＋ 64 ※1）×（Pi ＋ 4）<br>＋（124 ＋ 80 ※2 ＋ 96 × A × Mi）× Ui} ÷ 4096 ↑ × 4096<br>＋ Si × {Pi ＋ 4 ＋（Ui × Mi × A)}<br>} ÷ 1024<br>**● 64 ビットモードの場合**<br>$\sum_{i=1}^{n}$ {<br>↑ {928 ＋ 64 ＋（464 ＋ 112 ※1）×（Pi ＋ 4）<br>＋（176 ＋ 96 ※2 ＋ 136 × A × Mi）× Ui} ÷ 4096 ↑ × 4096<br>＋ Si × {Pi ＋ 4 ＋（Ui × Mi × A)}<br>} ÷ 1024<br>［訂正後］<br>**● 32 ビットモードの場合**<br>$\sum_{i=1}^{n}$ {《管理領域部》＋《データ格納部》} ÷ 1024<br>《管理領域部》:<br>↑ {736 ＋ 64 ＋（296 ＋ 64 ※1）×（Pi ＋ 4）<br>＋（124 ＋ 80 ※2 ＋ 96 × A × Mi）× Ui} ÷ 4096 ↑ × 4096<br>《データ格納部》:<br>Si × {Pi ＋ 4 ＋（Ui × Mi × A)}<br>**● 64 ビットモードの場合**<br>$\sum_{i=1}^{n}$ {《管理領域部》＋《データ格納部》} ÷ 1024<br>《管理領域部》:<br>↑ {928 ＋ 64 ＋（480 ＋ 112 ※1）×（Pi ＋ 4）<br>＋（176 ＋ 96 ※2 ＋ 136 × A × Mi）× Ui} ÷ 4096 ↑ × 4096<br>《データ格納部》:<br>Si × {Pi ＋ 4 ＋（Ui × Mi × A)} | 15.2.5 |

| 追加・変更内容 | 変更個所 |
| --- | --- |
| (1)(a) グローバルバッファが使用する共用メモリを実メモリ上に固定しない場合<br>表　計算式 2<br>［訂正前］<br>**● 32 ビットモードの場合**<br>{↑［((s × 1024 ÷ 4) ÷ 8) ＋ 112］÷ 4096 ↑× 4096 ×↑ a ÷ (s × 1024) ↑<br>} ÷ 1024<br>**● 64 ビットモードの場合**<br>{↑［((s × 1024 ÷ 4) ÷ 8) ＋ 144］÷ 4096 ↑× 4096 ×↑ a ÷ (s × 1024) ↑<br>} ÷ 1024<br>［訂正後］<br>**● 32 ビットモードの場合**<br>{↑［((s × 1024 ÷ 4) ÷ 8) ＋ 112］÷ 4096 ↑× 4096 ×↑ a ÷ (s × 1024) ↑<br>}<br>**● 64 ビットモードの場合**<br>{↑［((s × 1024 ÷ 4) ÷ 8) ＋ 144］÷ 4096 ↑× 4096 ×↑ a ÷ (s × 1024) ↑<br>} | 15.2.5 |
| (1)(b) グローバルバッファが使用する共用メモリを実メモリ上に固定する場合<br>表　計算式 1<br>［訂正前］<br>**● 32 ビットモードの場合**<br>n<br>Σ {<br>i=1<br>↑ {↑ {736 ＋ 64 ＋ (280 ＋ 64 ※1) × (Pi ＋ 4)<br>＋ (124 ＋ 80 ※2 ＋ 96 × A × Mi) × Ui} ÷ 4096 ↑× 4096<br>＋ Si × {Pi ＋ 4 ＋ (Ui × Mi × A)}} ÷ p ↑× p<br>} ÷ 1024<br>**● 64 ビットモードの場合**<br>n<br>Σ {<br>i=1<br>↑ {↑ {928 ＋ 64 ＋ (464 ＋ 112 ※1) × (Pi ＋ 4)<br>＋ (176 ＋ 96 ※2 ＋ 136 × A × Mi) × Ui} ÷ 4096 ↑× 4096<br>＋ Si × {Pi ＋ 4 ＋ (Ui × Mi × A)}} ÷ p ↑× p<br>} ÷ 1024<br>［訂正後］<br>**● 32 ビットモードの場合**<br>n<br>Σ {↑ {《管理領域部》＋《データ格納部》} ÷ p ↑× p} ÷ 1024<br>i=1<br>《管理領域部》:<br>↑ {↑ {736 ＋ 64 ＋ (296 ＋ 64 ※1) × (Pi ＋ 4)<br>＋ (124 ＋ 80 ※2 ＋ 96 × A × Mi) × Ui} ÷ 4096 ↑× 4096<br>《データ格納部》:<br>Si × {Pi ＋ 4 ＋ (Ui × Mi × A)}<br>**● 64 ビットモードの場合**<br>n<br>Σ {↑ {《管理領域部》＋《データ格納部》} ÷ p ↑× p} ÷ 1024<br>i=1<br>《管理領域部》:<br>↑ {↑ {928 ＋ 64 ＋ (480 ＋ 112 ※1) × (Pi ＋ 4)<br>＋ (176 ＋ 96 ※2 ＋ 136 × A × Mi) × Ui} ÷ 4096 ↑× 4096<br>《データ格納部》:<br>Si × {Pi ＋ 4 ＋ (Ui × Mi × A)} | 15.2.5 |

| 追加・変更内容 | 変更個所 |
|---|---|
| (1)(b) グローバルバッファが使用する共用メモリを実メモリ上に固定する場合<br>表　計算式 2<br>［訂正前］<br>**● 32 ビットモードの場合**<br>{↑ {↑ ［((s × 1024 ÷ 4) ÷ 8) ＋ 112］÷ 4096 ↑× 4096 ×↑ a ÷ (s × 1024) ↑} ÷ p ↑× p} ÷ 1024<br>**● 64 ビットモードの場合**<br>{↑ {↑ ［((s × 1024 ÷ 4) ÷ 8) ＋ 144］÷ 4096 ↑× 4096 ×↑ a ÷ (s × 1024) ↑} ÷ p ↑× p} ÷ 1024<br>［訂正後］<br>**● 32 ビットモードの場合**<br>{↑ {↑ ［((s × 1024 ÷ 4) ÷ 8) ＋ 112］÷ 4096 ↑× 4096 ×↑ a ÷ (s × 1024) ↑} ÷ p ↑× p}<br>**● 64 ビットモードの場合**<br>{↑ {↑ ［((s × 1024 ÷ 4) ÷ 8) ＋ 144］÷ 4096 ↑× 4096 ×↑ a ÷ (s × 1024) ↑} ÷ p ↑× p} | 15.2.5 |
| (2)(a) グローバルバッファが使用する共用メモリを実メモリ上に固定しない場合<br>［訂正前］<br>**● 32 ビットモードの場合**<br>$\sum_{i=1}^{n}$ {<br>(96 ＋ ((736 × (A ＋ B)) ＋ (272 × (F ＋ (8 × (A ＋ B))))<br>＋ 8 × F × (A ＋ B) ＋ 16) ＋ H ＋ D)<br>＋ 2048 ＋ G ＋ (E × F ＋ (8 × (A ＋ B)))<br>} ÷ 1024<br>**● 64 ビットモードの場合**<br>$\sum_{i=1}^{n}$ {<br>((144 ＋ ((928 × (A ＋ B)) ＋ (448 × (F ＋ (8 × (A ＋ B))))<br>＋ (16 × F × (A ＋ B)))) +16 ＋ H ＋ D)<br>＋ 2048 ＋ G ＋ (E × F ＋ (8 × (A ＋ B)))<br>} ÷ 1024<br>［訂正後］<br>**● 32 ビットモードの場合**<br>$\sum_{i=1}^{n}$ {《管理領域部》＋《データ格納部》} ÷1024<br>《管理領域部》:<br>((96 ＋ ((736 × (A ＋ B)) ＋ (288 × (F ＋ (8 × (A ＋ B))))<br>＋ (8 × F × (A ＋ B))) ＋ 16) ＋ H ＋ D)<br>《データ格納部》:<br>2048 ＋ G ＋ (E × F ＋ (8 × (A ＋ B)))<br>**● 64 ビットモードの場合**<br>$\sum_{i=1}^{n}$ {《管理領域部》＋《データ格納部》} ÷1024<br>《管理領域部》:<br>((144 ＋ ((928 × (A ＋ B)) ＋ (464 × (F ＋ (8 × (A ＋ B))))<br>＋ (16 × F × (A ＋ B)))) +16 ＋ H ＋ D)<br>《データ格納部》:<br>2048 ＋ G ＋ (E × F ＋ (8 × (A ＋ B))) | 15.2.5 |

| 追加・変更内容 | 変更個所 |
|---|---|
| (2)(b) グローバルバッファが使用する共用メモリを実メモリ上に固定する場合<br>［訂正前］<br>**● 32 ビットモードの場合**<br>n<br>Σ {<br>i=1<br>{↑（96 ＋（(736 ×（A ＋ B)）＋（272 ×（F ＋（8 ×（A ＋ B))))<br>＋ 8 × F ×（A ＋ B）＋ 16）＋ H ＋ D)<br>＋ 2048 ＋ G ＋（E × F ＋（8 ×（A ＋ B)））÷ p ↑} × p<br>} ÷ 1024<br><br>**● 64 ビットモードの場合**<br>n<br>Σ {<br>i=1<br>{↑ ((144 ＋（(928 ×（A ＋ B)）＋（448 ×（F ＋（8 ×（A ＋ B))))<br>＋（16 × F ×（A ＋ B)))) +16 ＋ H ＋ D)<br>＋ 2048 ＋ G ＋（E × F ＋（8 ×（A ＋ B)））÷ p ↑} × p<br>} ÷ 1024<br>［訂正後］<br>**● 32 ビットモードの場合**<br>n<br>Σ {↑ {《管理領域部》＋《データ格納部》} ÷ p ↑× p} ÷ 1024<br>i=1<br>《管理領域部》:<br>((96 ＋（(736 ×（A ＋ B)）＋（288 ×（F ＋（8 ×（A ＋ B))))<br>＋（8 × F ×（A ＋ B)））＋ 16）＋ H ＋ D)<br>《データ格納部》:<br>＋ 2048 ＋ G ＋（E × F ＋（8 ×（A ＋ B)））<br>**● 64 ビットモードの場合**<br>n<br>Σ {↑ {《管理領域部》+《データ格納部》} ÷p↑×p} ÷1024<br>i=1<br><br>《管理領域部》:<br>((144 ＋（(928 ×（A ＋ B)）＋（464 ×（F ＋（8 ×（A ＋ B))))<br>＋（16 × F ×（A ＋ B)））+16）＋ H ＋ D)<br>《データ格納部》:<br>2048 ＋ G ＋（E × F ＋（8 ×（A ＋ B)）) | 15.2.5 |
| 計算式　マスタディレクトリ用 RD エリアの容量（単位：バイト）<br>［訂正前］<br>(前略)<br>＋↑（d ＋ 240）÷ 6000 ↑× 51 ＋↑ e ÷ 6400 ↑× 51<br>(後略)<br>［訂正後］<br>(前略)<br>＋↑（d ＋ 240）÷ 64000 ↑× 51 ＋↑ e ÷ 64000 ↑× 51<br>(後略) | 16.3 |
| (3) S の求め方<br>(a) HiRDB/ シングルサーバの場合<br>S の計算式<br>［訂正前］<br>＋↑ 8192 ÷ g ↑＋↑（8192 − 128）÷ g ↑× {((a + m）÷ k）− 1}<br>［訂正後］<br>＋↑ 8192 ÷ g ↑＋↑（8192 − 128）÷ g ↑× {↑（a ＋ m）÷ k ↑− 1} | 17.3 |

| 追加・変更内容 | 変更個所 |
| --- | --- |
| ［訂正前］<br>リソース数に関連する環境変数に設定する値の見積もり式を次の表に示します。計算結果が環境変数の省略値より大きい場合は，その環境変数に計算結果を設定してください。設定しないと HiRDB を開始できない，又はユティリティを実行できないなどの問題が発生します。<br>［訂正後］<br>リソース数に関連する環境変数の設定について説明します。これらの環境変数を設定することで，Windows にはないメッセージキュー，セマフォ，及び共用メモリの機能が使えるようになります。<br>環境変数の設定値の見積もりはユニット単位で行います。ただし，設定方法には OS 単位とユニット単位の二つがあります。HiRDB の構成に応じて設定方法を選択してください。<br>• **OS 単位で設定する**<br>Windows のシステム環境変数として設定します。この場合，OS 内のすべてのユニットに対して同じ値が設定されます。OS 内に複数のユニットが存在するマルチ HiRDB 構成の場合は，各ユニットの見積もりの最大値を設定します。そのため，最大値を必要としないユニットはむだなリソースを使用することになります。<br>OS 内に一つのユニットしか存在しない場合は OS 単位の設定で問題ありません。<br>設定を有効にするためには，OS を再起動する必要があります。<br>• **ユニット単位で設定する**<br>pdntenv コマンドの -sr オプションで設定します。OS 内に複数のユニットが存在する場合，各ユニットに適した値を設定できます。<br>設定を有効にするためには，サービスを再開始する必要があります。<br>なお，両方で設定した場合，ユニット単位の設定値が優先されます。<br>(1) 見積もり式<br>見積もり式を次の表に示します。 | 20.1 |

| 追加・変更内容 | 変更個所 |
|---|---|
| 表 20-1　リソース数に関連する環境変数に設定する値（HiRDB/ シングルサーバの場合）<br>［訂正前］<br>［訂正後］ | 20.1 |

［訂正前］

| 種別 | 計算式 | 対応する環境変数 | 省略値 |
|---|---|---|---|
| メッセージキュー識別子数 | （16＋f）×a＋28＋1 | PDUXPLMSGMNI | 50 |
| メッセージキューテーブル数 | MAX（b×ユニット数，pdbufferの-m指定値の合計値） | PDUXPLMSGTQL | 80 |
| セマフォ識別子数 | ｛↑｛2×（b＋3）＋12｝÷64↑＋↑c÷64↑＋g＋5｝×a＋2＋d | PDUXPLSEMMAX | 64 |
| 共用メモリ使用数 | 2×e＋16 | PDUXPLSHMMAX | 4,096 |

［訂正後］

| 種別 | 計算式 | 対応する環境変数 | 設定範囲 | 省略値 |
|---|---|---|---|---|
| メッセージキュー識別子数（単位：個） | （16＋f）×a＋28＋1 | OS単位：システム環境変数PDUXPLMSGMNI<br>ユニット単位：pdntenv -sr msgmni | 50～3,600,000※1，2 | ●32ビットモードの場合<br>50<br>●64ビットモードの場合<br>100 |
| メッセージキューテーブル数（単位：個） | b×ユニット数＋A＋B | OS単位：システム環境変数PDUXPLMSGTQL<br>ユニット単位：pdntenv -sr msgtql | 80～2,048※1，2 | ●32ビットモードの場合<br>80<br>●64ビットモードの場合<br>2,048 |
| セマフォ識別子数（単位：個） | ｛↑｛2×（b＋3）＋12｝÷64↑＋↑c÷64↑＋g＋5｝×a＋2＋d | OS単位：システム環境変数PDUXPLSEMMAX<br>ユニット単位：pdntenv -sr semmax | 64～2,147,483,647※1 | 64 |
| 共用メモリ使用数（単位：セグメント） | （2＋h＋pd_max_add_dbbuff_shm_noの値※3＋pd_max_resident_rdarea_shm_noの値※4）×（e＋i＋50） | OS単位：システム環境変数PDUXPLSHMMAX<br>ユニット単位：pdntenv -sr shmmax | 4,096～2,147,483,647※1 | ●32ビットモードの場合<br>4,096<br>●64ビットモードの場合<br>16,384 |

| 追加・変更内容 | 変更個所 |
|---|---|
| 表 20-1　リソース数に関連する環境変数に設定する値（HiRDB/ シングルサーバの場合）<br>［追加］<br>注※ 1<br>設定範囲より小さい値を設定した場合，HiRDB が最小値に切り上げます。<br>注※ 2<br>設定範囲より大きい値を設定した場合，HiRDB が最大値に切り下げます。<br>注※ 3<br>グローバルバッファの動的変更機能を使用する場合に加算してください。セキュリティ監査機能使用時はさらに 1 を加算してください。<br>注※ 4<br>インメモリデータ処理を行う場合に加算してください。 | 20.1 |

| 追加・変更内容 | 変更個所 |
| --- | --- |
| 表 20-1　リソース数に関連する環境変数に設定する値（HiRDB/ シングルサーバの場合）<br>種別：メッセージキューテーブル数の「計算式」<br>［訂正前］<br>MAX（b ×ユニット数，pdbuffer の -m 指定値の合計値）<br>［訂正後］<br>b ×ユニット数＋ A ＋ B<br><br>A：次の値を代入してください。<br><br>$\sum_{i=1}^{N}$ {<br>サーバiに割り当てたグローバルバッファプール数※1<br>＋サーバiのシンクポイントダンプ有効化のスキップ回数※2×2<br>＋サーバiで実行するpdload，pdrorg，pdrbal，ログレスUAPの最大同時実行数＋1<br>}<br><br>N：1<br><br>注※1<br>pdbuflsコマンドで表示されるバッファプール名をサーバ単位に集計することで確認できます。<br>注※2<br>pd_spd_syncpoint_skip_limitに0以外の値を指定している場合はpd_spd_syncpoint_skip_ limit指定値で見積もります。<br>pd_spd_syncpoint_skip_limitを省略しているか，0を指定している場合は，マニュアル「HiRDB Version 8 システム運用ガイド」「8.8　UAPの状態監視　(3)計算方法　(b)全システムログファイルの容量から計算する方法」を参照し見積もってください。<br><br>B：非同期 READ 機能使用時（pd_max_ard_process に 0 以外指定時）だけ加算します。非同期 READ 機能を使用しない場合は 0 で見積もります。次の値を代入してください<br><br>$\sum_{i=1}^{N}$ {<br>サーバiに割り当てたグローバルバッファプールのpdbuffer-m指定値の合計<br>}<br><br>N：1 | 20.1 |
| 表 20-1　リソース数に関連する環境変数に設定する値（HiRDB/ シングルサーバの場合）<br>種別：共用メモリ使用数の「計算式」<br>［訂正前］<br>2 × e ＋ 16<br>［訂正後］<br>（2 ＋ h ＋ pd_max_add_dbbuff_shm_no の値※3 ＋ pd_max_resident_rdarea_shm_no の値※4）×（e ＋ i ＋ 50）<br>h：↑（グローバルバッファが使用する共用メモリの総量（※）÷ SHMMAX の値）↑<br>i：pd_aud_file_name オペランドを指定している場合は b の値，pd_aud_file_name オペランドを指定していない場合は 0<br>注※<br>「15.1.5　グローバルバッファが使用する共用メモリの計算式」を参照してください。 | 20.1 |

<table>
<tr><th>追加・変更内容</th><th>変更個所</th></tr>
<tr><td>［追加］<br>(2) 共用メモリの計算式<br>メッセージキュー，セマフォ，及び共用メモリの機能を使うことによって使用される共用メモリの計算式を次の表に示します。<br>表 20-2　共用メモリの計算式<br>
<table>
<tr><th>共用メモリを使う機能</th><th>計算式（単位：バイト）</th></tr>
<tr><td>メッセージキュー</td><td>●32ビットモードの場合<br>（（（（16384＋（メッセージキューテーブル数×24））＋15）÷16）×16＋384）<br>＋（（メッセージキュー識別子数－1）×64）<br>＋（（メッセージキューテーブル数－1）×32）<br>●64ビットモードの場合<br>（（（（16384＋（PDUXPLMSGTQL×24））＋15）÷16）×16＋408）<br>＋（（メッセージキュー識別子数 －1）×80）<br>＋（（メッセージキューテーブル数 －1）×40）</td></tr>
<tr><td>セマフォ</td><td>●32ビットモードの場合<br>64＋セマフォ識別子数の設定値×176<br>●64ビットモードの場合<br>104＋セマフォ識別子数の設定値×184</td></tr>
<tr><td>共用メモリ（共用メモリを使用する場合，管理用の共用メモリが必要になる）</td><td>●32ビットモードの場合<br>16＋共用メモリ使用数の設定値×40<br>●64ビットモードの場合<br>24＋共用メモリ使用数の設定値×64</td></tr>
</table>
</td><td>20.1</td></tr>
<tr><td>［訂正前］<br>リソース数に関連する環境変数に設定する値の見積もり式を次の表に示します。計算結果が環境変数の省略値より大きい場合は，その環境変数に計算結果を設定してください。設定しないと HiRDB を開始できない，又はユティリティを実行できないなどの問題が発生します。<br>なお，この計算式は一つのサーバマシン内で必要な値です。<br>［訂正後］<br>リソース数に関連する環境変数の設定について説明します。これらの環境変数を設定することで，Windows にはないメッセージキュー，セマフォ，及び共用メモリの機能が使えるようになります。<br>環境変数の設定値の見積もりはユニット単位で行います。ただし，設定方法には OS 単位とユニット単位の二つがあります。HiRDB の構成に応じて設定方法を選択してください。<br>• OS 単位で設定する<br>Windows のシステム環境変数として設定します。この場合，OS 内のすべてのユニットに対して同じ値が設定されます。OS 内に複数のユニットが存在するマルチ HiRDB 構成の場合は，各ユニットの見積もりの最大値を設定します。そのため，最大値を必要としないユニットはむだなリソースを使用することになります。<br>OS 内に一つのユニットしか存在しない場合は OS 単位の設定で問題ありません。<br><br>設定を有効にするためには，OS を再起動する必要があります。<br>• ユニット単位で設定する<br>pdntenv コマンドの -sr オプションで設定します。OS 内に複数のユニットが存在する場合，各ユニットに適した値を設定できます。<br>設定を有効にするためには，OS を再起動する必要があります。<br><br>なお，両方で設定した場合，ユニット単位の設定値が優先されます。<br>(1) 見積もり式<br>見積もり式を次の表に示します。</td><td>20.2</td></tr>
</table>

| 追加・変更内容 | 変更個所 |
|---|---|
| 表 20-3　リソース数に関連する環境変数に設定する値（HiRDB/パラレルサーバの場合）<br>［訂正前］<br><table><tr><th>種別</th><th>計算式</th><th>対応する環境変数</th><th>省略値</th></tr><tr><td>メッセージキュー識別子数</td><td>$\sum_{i=1}^{b} vi+2\times a+3\times b+c+d+e+24+1+m$</td><td>PDUXPLMSGMNI</td><td>50</td></tr><tr><td>メッセージキューテーブル数</td><td>MAX（h×ユニット数，pdbufferの-m指定値の合計値）</td><td>PDUXPLMSGTQL</td><td>80</td></tr><tr><td>セマフォ識別子数</td><td>●影響分散スタンバイレス型系切り替え機能を使用していない場合<br>（後略）</td><td>PDUXPLSEMMAX</td><td>64</td></tr><tr><td>共用メモリ使用数</td><td>2×g+16×b</td><td>PDUXPLSHMMAX</td><td>4,096</td></tr></table><br>［訂正後］<br><table><tr><th>種別</th><th>計算式</th><th>対応する環境変数</th><th>設定範囲</th><th>省略値</th></tr><tr><td>メッセージキュー識別子数<br>（単位：個）</td><td>$\sum_{i=1}^{b} vi+2\times a+3\times b+c+d+e+25+1+m$</td><td>OS単位：システム環境変数PDUXPLMSGMNI<br>ユニット単位：pdntenv -sr msgmni</td><td>50～3,600,000<br>※1，2</td><td>●32ビットモードの場合<br>50<br>●64ビットモードの場合<br>100</td></tr><tr><td>メッセージキューテーブル数<br>（単位：個）</td><td>h×ユニット数+A+B</td><td>OS単位：システム環境変数PDUXPLMSGTQL<br>ユニット単位：pdntenv -sr msgtql</td><td>80～2,048<br>※1，2</td><td>●32ビットモードの場合<br>80<br>●64ビットモードの場合<br>2,048</td></tr><tr><td>セマフォ識別子数<br>（単位：個）</td><td>●影響分散スタンバイレス型系切り替え機能を使用していない場合<br>（後略）</td><td>OS単位：システム環境変数PDUXPLSEMMAXユニット単位：pdntenv -sr semmax</td><td>64～2,147,483,647<br>※1</td><td>64</td></tr><tr><td>共用メモリ使用数<br>（単位：セグメント）</td><td>$(2+z+\sum_{i=1}^{n}$各サーバに指定したPd_max_add_dbbuff_shm_noの値※3+（pd_max_resident_rdarea_shm_noの値×バックエンドサーバ数）※4）×（g+Ai+50）</td><td>OS単位：システム環境変数PDUXPLSHMMAX<br>ユニット単位：pdntenv -sr shmmax</td><td>4,096～2,147,483,647<br>※1</td><td>●32ビットモードの場合<br>4,096<br>●64ビットモードの場合<br>16,384</td></tr></table> | 20.2 |
| 表 20-3　リソース数に関連する環境変数に設定する値（HiRDB/パラレルサーバの場合）<br>［追加］<br>注※ 1<br>設定範囲より小さい値を設定した場合，HiRDB が最小値に切り上げます。<br>注※ 2<br>設定範囲より大きい値を設定した場合，HiRDB が最大値に切り下げます。<br>注※ 3<br>グローバルバッファの動的変更機能を使用する場合に加算してください。セキュリティ監査機能使用時はさらに 1 を加算してください。<br>注※ 4<br>インメモリデータ処理を行う場合に加算してください。 | 20.2 |

| 追加・変更内容 | 変更個所 |
| --- | --- |
| 表 20-3　リソース数に関連する環境変数に設定する値（HiRDB/ パラレルサーバの場合）<br>［訂正前］<br>MAX（h ×ユニット数，pdbuffer の -m 指定値の合計値）<br>［訂正後］<br>h ×ユニット数＋ A ＋ B<br>A：次の値を代入してください。<br>$\sum_{i=1}^{N}$ {<br>サーバiに割り当てたグローバルバッファプール数※1<br>＋サーバiのシンクポイントダンプ有効化のスキップ回数※2×2<br>＋サーバiで実行するpdload，pdrorg，pdrbal，ログレスUAPの最大同時実行数＋1<br>}<br>N：<br>ユニット内のバックエンドサーバ数＋ユニット内のディクショナリサーバ数＋ユニット内のゲストBES数※3<br>注※1<br>pdbuflsコマンドで表示されるバッファプール名をサーバ単位に集計することで確認できます。<br>注※2<br>pd_spd_syncpoint_skip_limitに0以外の値を指定している場合はpd_spd_syncpoint_skip_ limit指定値で見積もります。<br>pd_spd_syncpoint_skip_limitを省略しているか，0を指定している場合は，マニュアル「HiRDB Version 8 システム運用ガイド」「8.8　UAPの状態監視　(3)計算方法　(b)全システムログファイルの容量から計算する方法」を参照し見積もってください。<br>注※3<br>影響分散スタンバイレス型系切り替え適用ユニットの場合に加算します。<br>B：非同期 READ 機能使用時（pd_max_ard_process に 0 以外指定時）だけ加算します。非同期 READ 機能を使用しない場合は 0 で見積もります。次の値を代入してください。<br>$\sum_{i=1}^{N}$ {<br>サーバiに割り当てたグローバルバッファプールのpdbuffer-m指定値の合計<br>}<br>N：<br>ユニット内のバックエンドサーバ数＋ユニット内のディクショナリサーバ数＋ユニット内のゲストBES数※<br>注※<br>影響分散スタンバイレス型系切り替え適用ユニットの場合に加算します。 | 20.2 |

<table>
<tr><th>追加・変更内容</th><th>変更個所</th></tr>
<tr><td>表 20-3　リソース数に関連する環境変数に設定する値（HiRDB/ パラレルサーバの場合）<br>種別：共用メモリ使用数の「計算式」<br>［訂正前］<br>2 × g ＋ 16 × b<br>［訂正後］<br>(2 ＋ z<br>n<br>＋ Σ 各サーバに指定した pd_max_add_dbbuff_shm_no の値※3<br>i=1<br>＋（pd_max_resident_rdarea_shm_no の値×バックエンドサーバ数）※4）×（g ＋ Ai ＋ 50）<br>z：↑（ユニット内のグローバルバッファが使用する共用メモリの総量（※）÷ SHMMAX の値）↑<br>A：pd_aud_file_name オペランドを指定している場合は h の値，pd_aud_file_name オペランドを指定していない場合は 0<br>注※<br>「15.2.5　グローバルバッファが使用する共用メモリの計算式」を参照してください。</td><td>20.2</td></tr>
<tr><td>［追加］<br>(2) 共用メモリの計算式<br>メッセージキュー，セマフォ，及び共用メモリの機能を使うことによって使用される共用メモリの計算式を次の表に示します。<br>表 20-4　共用メモリの計算式
<table>
<tr><th>共用メモリを使う機能</th><th>計算式（単位：バイト）</th></tr>
<tr><td>メッセージキュー</td><td>●32ビットモードの場合<br>（（（（16384＋（メッセージキューテーブル数×24））＋15）÷16）×16＋384）<br>＋（（メッセージキュー識別子数－1）×64）<br>＋（（メッセージキューテーブル数－1）×32）<br>●64ビットモードの場合<br>（（（（16384＋（メッセージキューテーブル数×24））＋15）÷16）×16＋408）<br>＋（（メッセージキュー識別子数 －1）×80）<br>＋（（メッセージキューテーブル数 －1）×40）</td></tr>
<tr><td>セマフォ</td><td>●32ビットモードの場合<br>64＋セマフォ識別子数の設定値×176<br>●64ビットモードの場合<br>104＋セマフォ識別子数の設定値×184</td></tr>
<tr><td>共用メモリ（共用メモリを使用する場合，管理用の共用メモリが必要になる）</td><td>●32ビットモードの場合<br>16＋共用メモリ使用数の設定値×40<br>●64ビットモードの場合<br>24＋共用メモリ使用数の設定値×64</td></tr>
</table></td><td>20.1</td></tr>
<tr><td>表 A-1　HiRDB のシステム構成に関する最大値と最小値<br>［訂正前］<br>データディクショナリ用 RD エリア数 ｜ 1 ｜ 41 ｜ 個 ｜<br>［訂正後］<br>データディクショナリ用 RD エリア数 ｜ 1 ｜ 59 ｜ 個 ｜<br>解析情報表及び運用履歴表を格納する ｜ 1 ｜ 1 ｜ 個 ｜<br>データディクショナリ用 RD エリア数</td><td>付録 A.1</td></tr>
</table>

**変更内容** (3020-6-352-40)　HiRDB Version 8 08-05

| 追加・変更内容 | 変更個所 |
|---|---|
| HiRDB.NET データプロバイダでメソッドトレースを取得できるようにしました。これに伴い，単調増加ファイルに HiRDB.NET データプロバイダのメソッドトレースを追加しました。 | 1.2.2(9) |
| Windows Server 2008 のクラスタソフトウェア（MSFC）の説明を追加しました。 | 1.4，1.4.7，10.2(4)(a)，10.2(4)(b) |

| 追加・変更内容 | 変更個所 |
|---|---|
| システムログファイルの空き容量不足を検知した場合，HiRDB が自動的にシステムログファイルを拡張できるようにしました（システムログファイルの自動拡張機能）。 | 3.1.1，3.3.1，3.6.2(4)(a)，8.2.4(1)，8.3.1(1)，9.2.4(1)，9.3.1(1) |
| 簡易セットアップツールを実行するマシン環境についての説明を追加しました。 | 3.1.4(2) |
| HiRDB/ シングルサーバの場合，HiRDB サーバに対する最大同時接続数を 3,000 まで拡大しました。これによって，より多くのユーザがデータベースに同時にアクセスできるようになりました。 | 3.4.1(1)，表 A-1 |
| pd_large_file_use オペランドの省略値を N から Y に変更しました。 | 4.3.1，8.2.6，9.2.6，15.1.3(1)，15.1.3(2)，15.2.3(1)，15.2.3(2) |
| raw I/O 機能で使用できるパーティションの形式について説明を追加しました。 | 4.3.2，4.3.2(2) |
| ビュー定義，および WITH 句の導出問合せ中に，抽象データ型の指定，およびそれらのビューや問合せを指定できるようにしました（プラグインが提供する抽象データ型を含みます）。 | 5.1.1(6)(b)，5.2，5.3(1) |
| データベース中でデータを呼び出すごとに一連の整数値を返す順序数生成子を追加しました（自動採番機能）。これに伴い，自動採番機能を使用したデータロードについての説明を追加しました。 | 6.1，6.1.7，表 A-1 |
| XML 型全文検索用インデクスの使用条件を追加しました。 | 6.4.2(4)，6.4.2(4)(d) |
| HiRDB が使用する共用メモリをメモリ上に固定できるようにしました。これによって，共用メモリのページングが防止できるため，アクセス性能が向上します。 | 8.1.1(1)，9.1.1(5) |
| バイナリデータの分岐行のページの LRU 管理を抑止できるようにしました。これによって，グローバルバッファにキャッシュされた内容がバイナリデータの読み込みによってメモリから追い出されるのを回避できます。 | 11.8.1(1)(c)，11.8.2，11.8.2(2)，11.8.2(4)，11.8.3 |
| 文字列データ型に文字集合 UTF-16 を指定できるようにしました。これによって，UTF-16 で作成された文字列データを，そのままの形式でデータベースに格納できるようになりました。 | 表 12-1，12.13(1)，12.13(2)，12.13(4) |
| 表の空き領域の再利用機能を使用している場合に，監視不要なセグメント使用率通知メッセージの出力を抑止できるようにしました。 | 14.5.5 |
| HiRDB/ シングルサーバのメモリ所要量の見積もり式を変更しました。 | 15.1.2，15.1.3(1)，15.1.3(2)，15.1.4，15.1.5(2)，15.1.10 |
| HiRDB/ パラレルサーバのメモリ所要量の見積もり式を変更しました。 | 15.2.2，15.2.3(1)，15.2.3(2)，15.2.4(1)，15.2.5(1)(b)，15.2.5(2)(b)，15.2.11 |
| ユーザ用 RD エリアの容量見積もりを変更しました。 | 16.1.1(2)，16.1.2(1)，16.1.2(3) |
| ディクショナリ表 SQL_RDAREAS の格納ページ数の見積もり式を変更しました。 | 16.2.1(2)(e) |
| ディクショナリ表 SQL_SEQUENCES の格納ページ数の見積もり式を追加しました。 | 16.2.1(2)(w)，表 16-6 |
| システムログファイルの容量の見積もり時の注意事項を追加しました。 | 17.1.1(3) |
| システムログファイルの容量の見積もりを変更しました。 | 17.1.3(1)，17.1.3(2)，17.1.4(2)(b)，17.1.9 |
| データベース状態解析ユティリティ（pddbst）実行時のファイルの容量を変更しました。 | 19.1.4 |
| 次に示すユティリティ実行時のメモリ所要量を変更しました。<br>• データベース作成ユティリティ（pdload）<br>• データベース再編成ユティリティ（pdrorg）<br>• データベース複写ユティリティ（pdcopy）<br>• データベース回復ユティリティ（pdrstr） | 19.2.3(3)，19.2.4(3)，19.2.9(1)，19.2.9(2)，19.2.10(1)，19.2.10(2) |

単なる誤字・脱字などはお断りなく訂正しました。

**変更内容** (3020-6-352-30)　HiRDB Version 8 08-04

| 追加・変更内容 |
|---|
| HiRDB の稼働プラットフォームに Windows Server 2008 を追加しました。 |
| HiRDB を使用すると単調増加するファイルについて，説明を追加しました。 |
| pdfmkfs コマンドで HiRDB ファイルシステム領域を作成する場合の，アクセス権の付与についての説明を追加しました。 |
| 簡易セットアップツールで，ウィザードによるセットアップができるようになりました。 |
| RD エリアの自動増分によって HiRDB ファイルシステム領域サイズの上限を超える場合，HiRDB ファイルシステム領域の上限を自動的に拡張するようにしました。 |
| 簡易セットアップツールで設定できる系切り替え機能について，説明を追加しました。 |
| XML 型を使用する場合に，次の機能が使えるようになりました。<br>• FLWOR 式の LET 句をサポートしました。また，XQuery 関数のサポート範囲を拡張しました。これによって，XML 型の値に対する問合せに記述できる XQuery 言語の範囲が広がりました。<br>• 全文検索用の hi-fn:contains 関数をサポートしました。これによって，XML 型の値に対して，異表記展開検索，および同義語検索などの全文検索機能が使用できるようになりました。<br>• XML 名前空間を使用した XML データが扱えるようになりました。 |
| XML 型の値を挿入または更新する場合，XML 変換コマンドや XML 変換ライブラリで XML 文書を ESIS-B 形式に変換する必要がありましたが，XML 文書を直接 INSERT 文や UPDATE 文に指定して値を挿入できるようにしました。 |
| マトリクス分割表の場合も，ALTER TABLE で分割格納条件を変更できるようにしました。 |
| 基本行を格納するセグメントと定義長 256 バイト以上の BINARY 列のデータを格納する BINARY 専用セグメントの情報を区別して表示するようにしました。これに伴い，説明を変更しました。 |
| オンライン更新業務実行中に共用 RD エリアのバックアップを取得できるようにしました。これによって，更新可能バックアップ閉塞時の制限を解除しました。 |
| HiRDB/ シングルサーバのメモリ所要量の見積もり式を変更しました。 |
| HiRDB/ パラレルサーバのメモリ所要量の見積もり式を変更しました。 |
| 次のディクショナリ表の格納ページ数の見積もり式を変更しました。<br>• SQL_COLUMN_STATISTICS<br>• SQL_VIEWS<br>• SQL_VIEW_DEF |
| レジストリ用 RD エリアの容量の見積もりを変更しました。 |
| ステータスファイルの容量の見積もりを変更しました。 |
| 次に示すユティリティで使用するファイルの容量の計算式を変更しました。<br>• データベース作成ユティリティ（pdload）<br>• データベース再編成ユティリティ（pdrorg）<br>• 統計解析ユティリティ（pdstedit）<br>• ディクショナリ搬出入ユティリティ（pdexp）<br>• リバランスユティリティ（pdrbal） |
| ディクショナリ搬出入ユティリティ（pdexp）実行時のメモリ所要量の計算式を変更しました。 |
| JP1/NETM/Audit と連携して，HiRDB が出力する監査証跡を JP1/NETM/Audit で一元管理できるようにしました。これに伴い，サンプルファイルに sampleAUDIT を追加しました。 |
| データベースに関する最大値と最小値を変更しました。 |

**変更内容** (3020-6-352-20)　HiRDB Version 8 08-03

| 追加・変更内容 |
|---|
| 次に示す条件をすべて満たす場合，hosts ファイルまたは DNS などにホスト名を登録する必要がなくなりました。<br>• HiRDB/ シングルサーバだけで HiRDB システムが構成されている<br>• システム定義およびクライアント環境定義中に指定するホスト名にループバックアドレスを指定している |
| 中国語漢字コード（GB18030）を使用できるようにしました。 |
| HiRDB が使用するポート番号やプログラムを Windows ファイアウォールの例外リストに登録する方法と削除する方法を記載しました。 |
| RD エリア内の全データをメモリ常駐化できるインメモリデータ処理をサポートしました。この機能を使用すると，バッチ処理の処理時間を短縮できます。 |
| 簡易セットアップツールが標準ホスト名の取得に失敗した場合に，［ローカルマシン名入力］画面でホスト名を入力できるようにしました。 |
| raw I/O 機能（ダイレクトディスクアクセス）を使用する場合の，HiRDB ファイルシステム領域を作成するときの制限（HiRDB ファイルシステム領域の使用目的に WORK，SVR を指定できない制限）を解除しました。 |
| HiRDB/ シングルサーバのメモリ所要量の見積もり式を変更しました。 |
| HiRDB/ パラレルサーバのメモリ所要量の見積もり式を変更しました。 |
| 次に示すディクショナリ表の格納ページ数の見積もり式を変更しました。<br>• SQL_VIEW_TABLE_USAGE<br>• SQL_ROUTINES<br>• SQL_ROUTINE_RESOURCES<br>• SQL_TRIGGER_USAGE |
| Java ルーチン，Java 手続き，および Java 関数という用語を，それぞれ外部 Java ルーチン，外部 Java 手続き，および外部 Java 関数という表記に変更しました。 |

**変更内容** (3020-6-352-10)　HiRDB Version 8 08-02

| 追加・変更内容 |
|---|
| HiRDB サーバおよび HiRDB クライアントの適用 OS に Windows Vista を追加しました。 |
| バージョン 08-02 以降から，バージョン 08-02 より前の HiRDB に戻す場合の説明を追加しました。 |
| インストール時に，システムの環境変数に HiRDB の環境変数を設定するかどうか選択できるようにしました。 |
| HiRDB が内部的に使用する作業用一時ファイルなどの出力先ディレクトリを指定できるようにしました。 |
| 簡易セットアップツールの稼働プラットフォームに Windows Vista を追加しました。 |
| 簡易セットアップツールで，詳細定義情報をローカルディスク上に保存したり，保存した詳細定義情報を読み込んだりできるようにしました。 |
| XML 形式で記述した文書を扱うデータ型である XML 型を追加しました。SQL 中に XML への問合せ宣言である XQuery を指定することによって，XML 型の値の部分構造の値を取り出したり，部分構造の値に対する条件評価をしたりできます。 |
| HiRDB/ パラレルサーバで 64 ユニットを超える大規模なシステムを構築する場合の説明を追加しました。 |
| 列定義，および埋込み変数の宣言で文字集合 EBCDIK を指定できるようにしました。 |
| ハッシュ分割をする場合，CREATE TABLE で表格納用 RD エリアを重複指定できるようにしました。これによって，各 RD エリアにデータをより均等に格納できるようになりました。 |
| ハッシュ分割で HASH0 を指定した場合で，範囲条件の指定があるときに，検索対象となる RD エリアを絞り込めるようにしました。 |
| HiRDB Control Manager を使用して，非改竄防止表を改竄防止表へ変更する例を記載しました。 |
| シングルサーバが使用する共用メモリの計算式を変更しました。 |
| HiRDB/ パラレルサーバの各サーバが使用する共用メモリの計算式を変更しました。 |
| データディクショナリ用 RD エリアの容量見積もりを変更しました。 |

| 追加・変更内容 |
| --- |
| レジストリ用 RD エリアの容量見積もりを変更しました。 |
| システムログファイルの容量の見積もりを変更しました。 |
| 監査証跡ファイルの容量見積もり式を追加しました。 |
| リソース数に関連する環境変数に設定する値の見積もりを変更しました。 |
| Windows デスクトップヒープ不足の発生を抑えるために，1 プロセス当たりのデスクトップヒープ消費量を指定できるようにしました。 |
| ファイアウォールを設置する場合に指定するオペランドの説明を変更しました。 |
| ユニットが使用する通信ポート数の見積もりを変更しました。 |
| HiRDB で指定するポート番号の説明を追加しました。 |
| HiRDB で起動するプロセスの一覧を追加しました。 |
| SANRISE という表記を日立ディスクアレイサブシステムに変更しました。 |

HiRDB Version 8 08-01

| 追加・変更内容 |
| --- |
| シングルサーバが使用する共用メモリの計算式を変更しました。 |
| HiRDB/ パラレルサーバの各サーバが使用する共用メモリの計算式を変更しました。 |
| ユーザ用 RD エリアの容量見積もりを変更しました。 |
| データディクショナリ用 RD エリアの容量見積もりを変更しました。 |
| システムログファイルの容量の見積もりを変更しました。 |
| ステータスファイルの容量の見積もりを変更しました。 |
| データベース作成ユティリティ（pdload）実行時のメモリ所要量の計算式を変更しました。 |
| データベース回復ユティリティ（pdrstr）実行時のメモリ所要量の計算式を変更しました。 |

# はじめに

このマニュアルは，プログラムプロダクト　スケーラブルデータベースサーバ　HiRDB Version 8 のシステムの構築方法，データベースの作成方法及びシステムとデータベースの設計方法について説明したものです。

## ■対象読者

HiRDB Version 8（以降，HiRDB と表記します）を使ってリレーショナルデータベースシステムを構築／運用する方々を対象にしています。
このマニュアルは次に示す知識があることを前提に説明しています。

- Windows のシステム管理の基礎的な知識
- SQL の基礎的な知識

また，このマニュアルは，マニュアル「HiRDB Version 8 解説」を前提としていますので，あらかじめお読みいただくことをお勧めします。

## ■マニュアルの構成

このマニュアルは，次に示す章と付録から構成されています。

**第 1 章　HiRDB のシステム構築の概要**
HiRDB のシステム構築手順，HiRDB のファイル構成及びバージョンアップ手順について説明しています。

**第 2 章　インストール**
インストール前に必要な作業，HiRDB のインストール手順，付加 PP インストール時の注意，及び HiRDB のアンインストールについて説明しています。
また，Windows ファイアウォールの例外リストへの登録方法と削除方法についても説明しています。

**第 3 章　簡易セットアップツールによる環境設定**
簡易セットアップツールを使用した HiRDB の環境設定について説明しています。

**第 4 章　コマンドによる環境設定**
コマンドを使用して HiRDB の環境を設定する方法について説明しています。

**第 5 章　プラグインの環境設定**
プラグインの環境設定方法，バージョンアップ方法及び削除（アンインストール）方法について説明しています。

**第 6 章　データベースの作成**
スキーマ，表，インデクスを作成し，データを格納するまでの方法について説明しています。

**第 7 章　ほかの製品との連携**
HiRDB とほかの製品とを連携する方法について説明しています。

**第 8 章　HiRDB/ シングルサーバの設計**
HiRDB/ シングルサーバのシステム構成，HiRDB ファイルシステム領域の設計方法，システムファイルの設計方法及び RD エリアの配置方法の考慮点について説明しています。

**第 9 章　HiRDB/ パラレルサーバの設計**
HiRDB/ パラレルサーバのシステム構成，HiRDB ファイルシステム領域の設計方法，システムファイルの設計方法及び RD エリアの配置方法の考慮点について説明しています。

**第 10 章　マルチ HiRDB の設計**
マルチ HiRDB のシステム設計で検討する項目について説明しています。

**第 11 章　グローバルバッファ，ローカルバッファの設計**

グローバルバッファ，ローカルバッファを設計する上で検討する項目について説明しています。

**第 12 章　表の設計**

表を設計する上で検討する項目について説明しています。

**第 13 章　インデクスの設計**

インデクス及びプラグインインデクスを設計する上で検討する項目について説明しています。

**第 14 章　RD エリアの設計**

RD エリアを構成するセグメント及びページを設計する上で検討する項目について説明しています。

**第 15 章　HiRDB のメモリ所要量**

HiRDB/ シングルサーバ及び HiRDB/ パラレルサーバのメモリ所要量の求め方について説明しています。

**第 16 章　RD エリアの容量の見積もり**

各 RD エリアの容量の見積もり方法について説明しています。

**第 17 章　システムファイル及び監査証跡ファイルの容量の見積もり**

システムログファイル，シンクポイントダンプファイル，ステータスファイルなどのシステムファイルの容量の見積もり方法，及び監査証跡ファイルの容量の見積もり方法について説明しています。

**第 18 章　作業表用ファイルの容量の見積もり**

作業表用ファイルの容量の見積もり方法について説明しています。

**第 19 章　ユティリティ実行時の容量の見積もり**

ユティリティ実行時のファイル容量及びメモリ所要量の見積もり方法について説明しています。

**第 20 章　リソース数に関連する環境変数の見積もり**

リソース数に関連する環境変数の見積もり方法について説明しています。

**第 21 章　Windows レジストリ設定値の見積もり**

Windows レジストリ設定値の見積もり方法について説明しています。

**第 22 章　サンプルファイル**

HiRDB が提供するサンプルファイル（サンプルデータベース，サンプルコンフィグレーション及びサンプル UOC）について説明しています。

**第 23 章　HiRDB サーバと HiRDB クライアント間の通信**

HiRDB サーバと HiRDB クライアントの接続方法，DNS サーバやファイアウォールがある場合の設定などについて説明しています。

**付録 A　HiRDB の最大値・最小値**

HiRDB のシステム構成の最大値及び最小値について説明しています。

**付録 B　HiRDB のプロセス一覧**

HiRDB で起動するプロセスについて説明しています。

**付録 C　Q&A**

HiRDB のシステム構築，設計及び運用時の注意について，Q&A 形式で説明しています。

**付録 D　バッチファイルによる環境設定**

サンプルの環境を自動的に設定するバッチファイルを使用した HiRDB サーバの環境設定方法について説明しています。

## ■関連マニュアル

このマニュアルの関連マニュアルを次に示します。必要に応じてお読みください。

HiRDB（Windows 用マニュアル）

- HiRDB Version 8 解説（Windows(R) 用）（3020-6-351）
- HiRDB Version 8 システム定義（Windows(R) 用）（3020-6-353）
- HiRDB Version 8 システム運用ガイド（Windows(R) 用）（3020-6-354）
- HiRDB Version 8 コマンドリファレンス（Windows(R) 用）（3020-6-355）
- HiRDB ファーストステップガイド（Windows(R) 用）（3020-6-054）

HiRDB（UNIX 用マニュアル）

- HiRDB Version 8 解説（UNIX(R) 用）（3000-6-351）
- HiRDB Version 8 システム導入・設計ガイド（UNIX(R) 用）（3000-6-352）
- HiRDB Version 8 システム定義（UNIX(R) 用）（3000-6-353）
- HiRDB Version 8 システム運用ガイド（UNIX(R) 用）（3000-6-354）
- HiRDB Version 8 コマンドリファレンス（UNIX(R) 用）（3000-6-355）
- インナレプリカ機能 HiRDB Staticizer Option Version 8（3000-6-363）
- HiRDB Version 8 ディザスタリカバリシステム　構築・運用ガイド（3000-6-364）
- HiRDB ファーストステップガイド（UNIX(R) 用）（3000-6-254）

HiRDB（Windows，UNIX 共通マニュアル）

- HiRDB Version 8 UAP 開発ガイド（3020-6-356）
- HiRDB Version 8 SQL リファレンス（3020-6-357）
- HiRDB Version 8 メッセージ（3020-6-358）
- HiRDB Version 8 セキュリティガイド（3020-6-359）
- HiRDB Version 8 XDM/RD E2 接続機能（3020-6-365）
- HiRDB Version 8 バッチ高速化機能（3020-6-368）
- HiRDB データ連動機能 HiRDB Datareplicator Version 8（3020-6-360）
- HiRDB データ連動拡張機能 HiRDB Datareplicator Extension Version 8（3020-6-361）
- データベース抽出・反映サービス機能 HiRDB Dataextractor Version 8（3020-6-362）
- HiRDB 全文検索プラグイン HiRDB Text Search Plug-in Version 8（3020-6-375）
- HiRDB XML 拡張機能 HiRDB XML Extension Version 8（3020-6-376）

なお，本文中で使用している HiRDB Version 8 のマニュアル名は，（UNIX(R) 用）または（Windows(R) 用）を省略して表記しています。使用しているプラットフォームに応じて UNIX 用または Windows 用のマニュアルを参照してください。

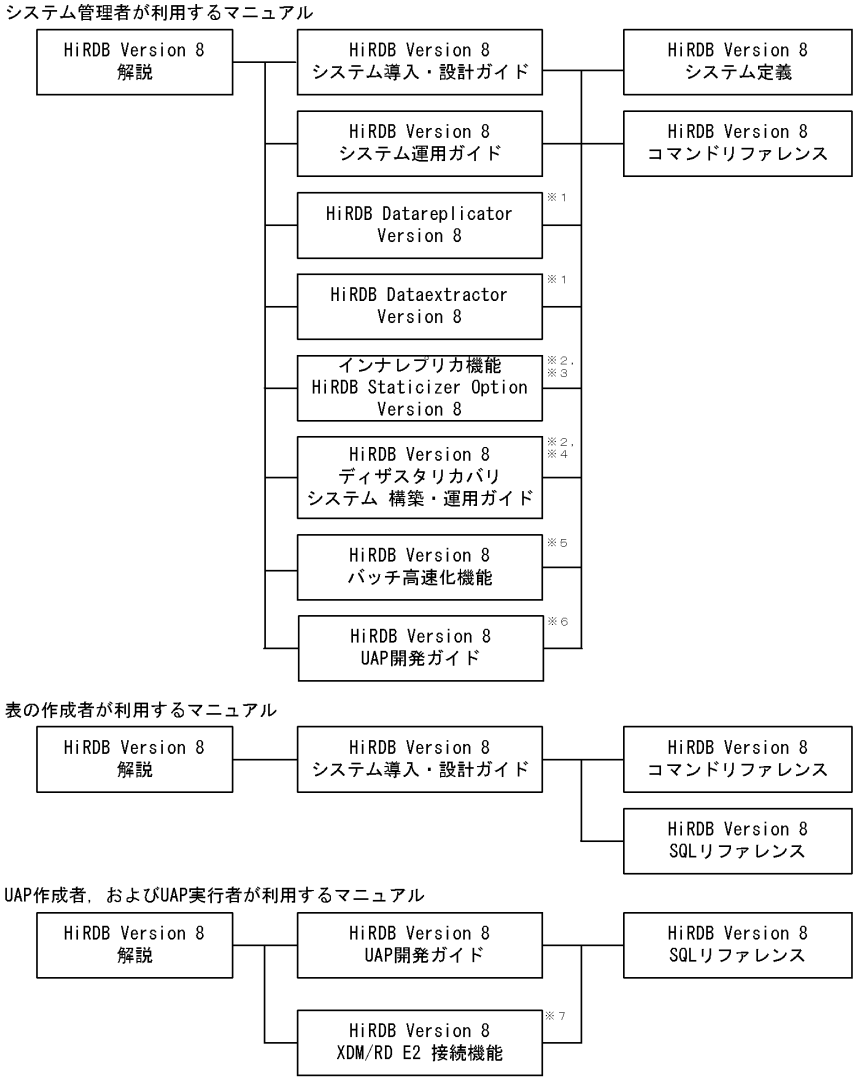
**関連製品**

- HiRDB External Data Access Version 8（3020-6-366）
- OpenTP1 Version 7 分散トランザクション処理機能　OpenTP1　プログラム作成の手引（3000-3-D51）
- OpenTP1 Version 7 分散トランザクション処理機能　OpenTP1　システム定義（3000-3-D52）
- JP1 Version 8 JP1/Integrated Management - Manager システム構築・運用ガイド（3020-3-K01）
- JP1 Version 8 JP1/Base 運用ガイド（3020-3-K06）
- JP1 Version 8 JP1/Automatic Job Management System 2 解説（3020-3-K21）
- JP1 Version 7i JP1/Integrated Manager - Console システム構築・運用ガイド（3020-3-F01）
- JP1 Version 7i JP1/Base（3020-3-F04）
- JP1 Version 7i JP1/Automatic Job Management System 2 解説（3020-3-F06）
- JP1 Version 6 JP1/Base（3020-3-986）
- JP1 Version 6 JP1/Automatic Job Management System 2 運用・操作編（3020-3-980）

- コード変換 ユーザーズガイド（3020-7-350）

## ■利用者ごとの関連マニュアル

HiRDB のマニュアルをご利用になる場合，利用者ごとに次のようにお読みください。
また，より理解を深めるために，左側のマニュアルから順にお読みいただくことをお勧めします。




注※１　レプリケーション機能を使用してデータ連携をする場合にお読みください。
注※２　UNIX用マニュアルです。Windows用はありません。
注※３　インナレプリカ機能を使用する場合にお読みください。
注※４　ディザスタリカバリシステムを構築する場合にお読みください。
注※５　インメモリデータ処理によるバッチ高速化を行う場合にお読みください。
注※６　OLTPシステムと連携する場合は必ずお読みください。
注※７　XDM/RD E2 接続機能を使用して，XDM/RD E2のデータベースを操作する場合にお読みください。

## ■読書手順

このマニュアルは，次に示す表に従ってお読みいただくことをお勧めします。

| 章タイトル | 参照の目安 |
|---|---|
| 1. HiRDB のシステム構築の概要 | ○ |
| 2. インストール | ○ |
| 3. 簡易セットアップツールによる環境設定 | ○ |
| 4. コマンドによる環境設定 | △ |
| 5. プラグインの環境設定 | △ |
| 6. データベースの作成 | ○ |
| 7. ほかの製品との連携 | △ |
| 8. HiRDB/ シングルサーバの設計 | ○ |
| 9. HiRDB/ パラレルサーバの設計 | ○ |
| 10. マルチ HiRDB の設計 | △ |
| 11. グローバルバッファ，ローカルバッファの設計 | ○ |
| 12. 表の設計 | ○ |
| 13. インデクスの設計 | ○ |
| 14. RD エリアの設計 | ○ |
| 15. HiRDB のメモリ所要量 | ○ |
| 16. RD エリアの容量の見積もり | ○ |
| 17. システムファイル及び監査証跡ファイルの容量の見積もり | ○ |
| 18. 作業表用ファイルの容量の見積もり | ○ |
| 19. ユティリティ実行時の容量の見積もり | ○ |
| 20. リソース数に関連する環境変数の見積もり | ○ |
| 21. Windows レジストリ設定値の見積もり | ○ |
| 22. サンプルファイル | △ |
| 23. HiRDB サーバと HiRDB クライアント間の通信 | △ |
| 付録 A HiRDB の最大値・最小値 | △ |
| 付録 B HiRDB のプロセス一覧 | △ |
| 付録 C Q&A | △ |
| 付録 D バッチファイルによる環境設定 | △ |

（凡例）

○：必ず読んでいただきたい内容です。

△：必要に応じて読んでいただきたい内容です。

## ■このマニュアルでの表記

このマニュアルでは製品名称および名称について次のように表記しています。ただし，それぞれのプログラムについての表記が必要な場合はそのまま表記しています。

| 製品名称または名称 | 表記 | |
|---|---|---|
| HiRDB/Single Server Version 8 | HiRDB/ シングルサーバ | HiRDB または HiRDB サーバ |
| HiRDB/Single Server Version 8(64) | | |
| HiRDB/Parallel Server Version 8 | HiRDB/ パラレルサーバ | |
| HiRDB/Parallel Server Version 8(64) | | |
| HiRDB/Developer's Kit Version 8 | HiRDB/Developer's Kit | HiRDB クライアント |
| HiRDB/Developer's Kit Version 8(64) | | |
| HiRDB/Run Time Version 8 | HiRDB/Run Time | |
| HiRDB/Run Time Version 8(64) | | |
| HiRDB Datareplicator Version 8 | HiRDB Datareplicator | |
| HiRDB Dataextractor Version 8 | HiRDB Dataextractor | |
| HiRDB Text Search Plug-in Version 8 | HiRDB Text Search Plug-in | |
| HiRDB XML Extension Version 8 | HiRDB XML Extension | |
| HiRDB Spatial Search Plug-in Version 3 | HiRDB Spatial Search Plug-in | |
| HiRDB Staticizer Option Version 8 | HiRDB Staticizer Option | |
| HiRDB LDAP Option Version 8 | HiRDB LDAP Option | |
| HiRDB Advanced Partitioning Option Version 8 | HiRDB Advanced Partitioning Option | |
| HiRDB Advanced High Availability Version 8 | HiRDB Advanced High Availability | |
| HiRDB Non Recover Front End Server Version 8 | HiRDB Non Recover FES | |
| HiRDB Disaster Recovery Light Edition Version 8 | HiRDB Disaster Recovery Light Edition | |
| HiRDB Accelerator Version 8 | HiRDB Accelerator | |
| HiRDB External Data Access Version 8 | HiRDB External Data Access | |
| HiRDB External Data Access Adapter Version 8 | HiRDB External Data Access Adapter | |
| HiRDB Adapter for XML - Standard Edition | HiRDB Adapter for XML | |
| HiRDB Adapter for XML - Enterprise Edition | | |
| HiRDB Control Manager | HiRDB CM | |
| HiRDB Control Manager Agent | HiRDB CM Agent | |
| Hitachi TrueCopy | TrueCopy | |
| Hitachi TrueCopy basic | | |
| TrueCopy | | |
| TrueCopy remote replicator | | |
| JP1/Automatic Job Management System 2 | JP1/AJS2 | |
| JP1/Automatic Job Management System 2 - Scenario Operation | JP1/AJS2-SO | |
| JP1/Cm2/Extensible SNMP Agent | JP1/ESA | |
| JP1/Cm2/Extensible SNMP Agent for Mib Runtime | | |
| JP1/Cm2/Network Node Manager | JP1/NNM | |
| JP1/Integrated Management - Manager | JP1/Integrated Management または JP1/IM | |
| JP1/Integrated Management - View | | |

| 製品名称または名称 | 表記 | |
|---|---|---|
| JP1/Magnetic Tape Access | EasyMT | |
| EasyMT | | |
| JP1/Magnetic Tape Library | MTguide | |
| JP1/NETM/Audit - Manager | JP1/NETM/Audit | |
| JP1/NETM/DM | JP1/NETM/DM | |
| JP1/NETM/DM Manager | | |
| JP1/Performance Management | JP1/PFM | |
| JP1/Performance Management - Agent Option for HiRDB | JP1/PFM-Agent for HiRDB | |
| JP1/Performance Management - Agent Option for Platform | JP1/PFM-Agent for Platform | |
| JP1/Performance Management/SNMP System Observer | JP1/SSO | |
| JP1/VERITAS NetBackup BS v4.5 | NetBackup | |
| JP1/VERITAS NetBackup v4.5 | | |
| JP1/VERITAS NetBackup BS V4.5 Agent for HiRDB License | JP1/VERITAS NetBackup Agent for HiRDB License | |
| JP1/VERITAS NetBackup V4.5 Agent for HiRDB License | | |
| JP1/VERITAS NetBackup 5 Agent for HiRDB License | | |
| OpenTP1/Server Base Enterprise Option | TP1/EE | |
| Virtual-storage Operating System 3/Forefront System Product | VOS3/FS | VOS3 |
| Virtual-storage Operating System 3/Leading System Product | VOS3/LS | |
| Extensible Data Manager/Base Extended Version 2<br>XDM 基本プログラム XDM/BASE E2 | XDM/BASE E2 | |
| XDM/Data Communication and Control Manager 3<br>XDM データコミュニケーションマネジメントシステム XDM/DCCM3 | XDM/DCCM3 | |
| XDM/Relational Database<br>リレーショナルデータベースシステム XDM/RD | XDM/RD | XDM/RD |
| XDM/Relational Database Extended Version 2<br>リレーショナルデータベースシステム XDM/RD E2 | XDM/RD E2 | |
| VOS3 Database Connection Server | DB コネクションサーバ | |
| BEA WebLogic Server | WebLogic Server | |
| DB2 Universal Database for OS/390 Version 6 | DB2 | |
| DNCWARE ClusterPerfect（Linux 版） | ClusterPerfect | |
| Microsoft(R) Office Excel | Microsoft Excel または Excel | |
| Microsoft(R) Visual C++(R) | Visual C++ または C++ 言語 | |
| Oracle8i | ORACLE | |
| Oracle9i | | |
| Oracle 10g | | |
| Sun Java™ System Directory Server | Sun Java System Directory Server または ディレクトリサーバ | |
| HP-UX 11i V2 (IPF) | HP-UX または HP-UX (IPF) | |
| HP-UX 11i V3 (IPF) | | |
| AIX 5L V5.1 | AIX 5L | AIX |
| AIX 5L V5.2 | | |

| 製品名称または名称 | 表記 | |
|---|---|---|
| AIX 5L V5.3 | | |
| AIX V6.1 | AIX V6.1 | |
| Linux(R) | Linux | |
| Red Hat Linux | Red Hat Linux | Linux |
| Red Hat Enterprise Linux | Red Hat Enterprise Linux | |
| Red Hat Enterprise Linux AS 3 (IPF) | Linux (IPF) | |
| Red Hat Enterprise Linux AS 4 (IPF) | | |
| Red Hat Enterprise Linux 5.1 Advanced Platform (Intel Itanium) | | |
| Red Hat Enterprise Linux 5.1 (Intel Itanium) | | |
| Red Hat Enterprise Linux 5.2 Advanced Platform (Intel Itanium) | | |
| Red Hat Enterprise Linux 5.2 (Intel Itanium) | | |
| Red Hat Enterprise Linux AS 3(AMD64 & Intel EM64T) | Linux (EM64T) | |
| Red Hat Enterprise Linux AS 4(AMD64 & Intel EM64T) | | |
| Red Hat Enterprise Linux ES 4(AMD64 & Intel EM64T) | | |
| Red Hat Enterprise Linux 5.1 Advanced Platform (AMD/Intel 64) | | |
| Red Hat Enterprise Linux 5.1 (AMD/Intel 64) | | |
| Red Hat Enterprise Linux 5.2 Advanced Platform (AMD/Intel 64) | | |
| Red Hat Enterprise Linux 5.2 (AMD/Intel 64) | | |
| Red Hat Enterprise Linux AS 4(AMD64 & Intel EM64T) | Linux AS 4 | |
| Red Hat Enterprise Linux AS 4(x86) | | |
| Red Hat Enterprise Linux ES 4(AMD64 & Intel EM64T) | Linux ES 4 | |
| Red Hat Enterprise Linux ES 4(x86) | | |
| Red Hat Enterprise Linux 5.1 Advanced Platform (x86) | Linux 5.1 | Linux 5 |
| Red Hat Enterprise Linux 5.1 (x86) | | |
| Red Hat Enterprise Linux 5.1 Advanced Platform (AMD/Intel 64) | | |
| Red Hat Enterprise Linux 5.1 (AMD/Intel 64) | | |
| Red Hat Enterprise Linux 5.1 Advanced Platform (Intel Itanium) | | |
| Red Hat Enterprise Linux ES 4(x86) | | |
| Red Hat Enterprise Linux 5.2 Advanced Platform (x86) | Linux 5.2 | |
| Red Hat Enterprise Linux 5.2 (x86) | | |
| Red Hat Enterprise Linux 5.2 Advanced Platform (AMD/Intel 64) | | |
| Red Hat Enterprise Linux 5.2 (AMD/Intel 64) | | |
| Red Hat Enterprise Linux 5.2 Advanced Platform (Intel Itanium) | | |
| Red Hat Enterprise Linux 5.2 (Intel Itanium) | | |
| turbolinux 7 Server for AP8000 | Linux for AP8000 | |
| Microsoft(R) Windows NT(R) Workstation Operating System Version 4.0 | Windows NT | |
| Microsoft(R) Windows NT(R) Server Network Operating System Version 4.0 | | |
| Microsoft(R) Windows(R) 2000 Professional Operating System | Windows 2000 | |
| Microsoft(R) Windows(R) 2000 Server Operating System | | |
| Microsoft(R) Windows(R) 2000 Datacenter Server Operating System | | |

| 製品名称または名称 | 表記 | |
|---|---|---|
| Microsoft(R) Windows(R) 2000 Advanced Server Operating System | | |
| Microsoft(R) Windows(R) 2000 Advanced Server Operating System | Windows 2000 Advanced Server | |
| Microsoft(R) Windows Server(R) 2003, Standard Edition | Windows Server 2003 Standard Edition | Windows Server 2003 |
| Microsoft(R) Windows Server(R) 2003, Enterprise Edition | Windows Server 2003 Enterprise Edition | |
| Microsoft(R) Windows Server(R) 2003, Standard x64 Edition | Windows Server 2003 Standard x64 Edition | |
| Microsoft(R) Windows Server(R) 2003, Enterprise x64 Edition | Windows Server 2003 Enterprise x64 Edition | |
| Microsoft(R) Windows Server(R) 2003 R2, Standard Edition | Windows Server 2003 R2 | |
| Microsoft(R) Windows Server(R) 2003 R2, Enterprise Edition | | |
| Microsoft(R) Windows Server(R) 2003 R2, Standard x64 Edition | | |
| Microsoft(R) Windows Server(R) 2003 R2, Enterprise x64 Edition | | |
| Microsoft(R) Windows Server(R) 2003 R2, Standard x64 Edition | Windows Server 2003 R2 x64 Editions | |
| Microsoft(R) Windows Server(R) 2003 R2, Enterprise x64 Edition | | |
| Microsoft(R) Windows Server(R) 2008 Standard | Windows Server 2008 Standard | Windows Server 2008 |
| Microsoft(R) Windows Server(R) 2008 Enterprise | Windows Server 2008 Enterprise | |
| Microsoft(R) Windows Server(R) 2008 R2 Standard (x64) | Windows Server 2008 R2 | |
| Microsoft(R) Windows Server(R) 2008 R2 Enterprise (x64) | | |
| Microsoft(R) Windows Server(R) 2008 R2 Datacenter (x64) | | |
| Microsoft(R) Windows Server(R) 2008 Standard (x64) | Windows Server 2008 (x64) | |
| Microsoft(R) Windows Server(R) 2008 Enterprise (x64) | | |
| Microsoft(R) Windows Server(R) 2003, Standard x64 Edition | Windows Server 2003 x64 Editions | Windows (x64) |
| Microsoft(R) Windows Server(R) 2003, Enterprise x64 Edition | | |
| Microsoft(R) Windows Server(R) 2003 R2, Standard x64 Edition | | |
| Microsoft(R) Windows Server(R) 2003 R2, Enterprise x64 Edition | | |
| Microsoft(R) Windows(R) XP Professional x64 Edition | Windows XP x64 Edition | |
| Microsoft(R) Windows Server(R) 2003, Enterprise x64 Edition | Windows Server 2003 (IPF) | Windows(IPF) |
| Microsoft(R) Windows(R) XP Professional x64 Edition | Windows XP x64 Edition | Windows XP |
| Microsoft(R) Windows(R) XP Professional Operating System | Windows XP Professional | |
| Microsoft(R) Windows(R) XP Home Edition Operating System | Windows XP Home Edition | |
| Microsoft(R) Windows Vista(R) Home Basic | Windows Vista Home Basic | Windows Vista |

| 製品名称または名称 | 表記 | |
|---|---|---|
| Microsoft(R) Windows Vista(R) Home Premium | Windows Vista Home Premium | |
| Microsoft(R) Windows Vista(R) Ultimate | Windows Vista Ultimate | |
| Microsoft(R) Windows Vista(R) Business | Windows Vista Business | |
| Microsoft(R) Windows Vista(R) Enterprise | Windows Vista Enterprise | |
| Microsoft(R) Windows Vista(R) Home Basic (x64) | Windows Vista (x64) | |
| Microsoft(R) Windows Vista(R) Home Premium (x64) | | |
| Microsoft(R) Windows Vista(R) Ultimate (x64) | | |
| Microsoft(R) Windows Vista(R) Business (x64) | | |
| Microsoft(R) Windows Vista(R) Enterprise (x64) | | |
| Microsoft(R) Windows Vista(R) Ultimate (x64) | Windows Vista Ultimate (x64) | |
| Microsoft(R) Windows Vista(R) Business (x64) | Windows Vista Business (x64) | |
| Microsoft(R) Windows Vista(R) Enterprise (x64) | Windows Vista Enterprise (x64) | |
| Microsoft(R) Windows(R) 7 Home Premium | Windows 7 Home Premium | Windows 7 |
| Microsoft(R) Windows(R) 7 Professional | Windows 7 Professional | |
| Microsoft(R) Windows(R) 7 Enterprise | Windows 7 Enterprise | |
| Microsoft(R) Windows(R) 7 Ultimate | Windows 7 Ultimate | |
| Microsoft(R) Windows(R) 7 Home Premium (x64) | Windows 7 (x64) | |
| Microsoft(R) Windows(R) 7 Professional (x64) | | |
| Microsoft(R) Windows(R) 7 Enterprise (x64) | | |
| Microsoft(R) Windows(R) 7 Ultimate (x64) | | |
| Microsoft(R) Windows(R) 7 Professional (x64) | Windows 7 Professional (x64) | |
| Microsoft(R) Windows(R) 7 Enterprise (x64) | Windows 7 Enterprise (x64) | |
| Microsoft(R) Windows(R) 7 Ultimate (x64) | Windows 7 Ultimate (x64) | |
| シングルサーバ | SDS | |
| システムマネジャ | MGR | |
| フロントエンドサーバ | FES | |
| ディクショナリサーバ | DS | |
| バックエンドサーバ | BES | |

- Windows Server 2003 および Windows Server 2008 を総称して Windows Server と表記します。また，Windows 2000，Windows XP，Windows Server，Windows Vista，および Windows 7 を総称して Windows と表記します。
- HiRDB 運用ディレクトリのパスを %PDDIR% と表記します。また，Windows のインストールディレクトリの

パスを %windir% と表記します。

- TCP/IP が規定する hosts ファイルを hosts ファイルと表記します。hosts ファイルとは通常，%windir%¥system32¥drivers¥etc¥hosts のことです。

## ■このマニュアルで使用する略語

このマニュアルで使用する英略語の一覧を次に示します。

| 英略語 | 英字の表記 |
|---|---|
| ACK | Acknowledgement |
| ADM | Adaptable Data Manager |
| ADO | ActiveX Data Objects |
| ADT | Abstract Data Type |
| AP | Application Program |
| API | Application Programming Interface |
| ASN.1 | Abstract Syntax Notation One |
| BES | Back End Server |
| BLOB | Binary Large Object |
| BMP | Basic Multilingual Plane |
| BOM | Byte Order Mark |
| CD-ROM | Compact Disc - Read Only Memory |
| CGI | Common Gateway Interface |
| CLOB | Character Large Object |
| CMT | Cassette Magnetic Tape |
| COBOL | Common Business Oriented Language |
| CORBA | Common ORB Architecture |
| CPU | Central Processing Unit |
| CSV | Comma Separated Values |
| DAO | Data Access Object |
| DAT | Digital Audio Taperecorder |
| DB | Database |
| DBM | Database Module |
| DBMS | Database Management System |
| DDL | Data Definition Language |
| DF for Windows NT | Distributing Facility for Windows NT |
| DF/UX | Distributing Facility / for UNIX |
| DIC | Dictionary Server |
| DLT | Digital Linear Tape |
| DML | Data Manipulate Language |
| DNS | Domain Name System |
| DOM | Document Object Model |

| 英略語 | 英字の表記 |
|---|---|
| DS | Dictionary Server |
| DTD | Document Type Definition |
| DTP | Distributed Transaction Processing |
| DWH | Data Warehouse |
| EUC | Extended UNIX Code |
| EX | Exclusive |
| FAT | File Allocation Table |
| FD | Floppy Disk |
| FES | Front End Server |
| FQDN | Fully Qualified Domain Name |
| FTP | File Transfer Protocol |
| GUI | Graphical User Interface |
| HBA | Host Bus Adapter |
| HD | Hard Disk |
| HTML | Hyper Text Markup Language |
| ID | Identification number |
| IP | Internet Protocol |
| IPF | Itanium(R) Processor Family |
| JAR | Java Archive File |
| Java VM | Java Virtual Machine |
| JDBC | Java Database Connectivity |
| JDK | Java Developer's Kit |
| JFS | Journaled File System |
| JFS2 | Enhanced Journaled File System |
| JIS | Japanese Industrial Standard code |
| JP1 | Job Management Partner 1 |
| JRE | Java Runtime Environment |
| JTA | Java Transaction API |
| JTS | Java Transaction Service |
| KEIS | Kanji processing Extended Information System |
| LAN | Local Area Network |
| LDAP | Lightweight Directory Access Protocol |
| LIP | Loop Initialization Process |
| LOB | Large Object |
| LRU | Least Recently Used |
| LTO | Linear Tape-Open |
| LU | Logical Unit |
| LUN | Logical Unit Number |

| 英略語 | 英字の表記 |
|---|---|
| LVM | Logical Volume Manager |
| MGR | System Manager |
| MIB | Management Information Base |
| MRCF | Multiple RAID Coupling Feature |
| MSCS | Microsoft Cluster Server |
| MSFC | Microsoft Failover Cluster |
| NAFO | Network Adapter Fail Over |
| NAPT | Network Address Port Translation |
| NAT | Network Address Translation |
| NIC | Network Interface Card |
| NIS | Network Information Service |
| NTFS | New Technology File System |
| ODBC | Open Database Connectivity |
| OLAP | Online Analytical Processing |
| OLE | Object Linking and Embedding |
| OLTP | On-Line Transaction Processing |
| OOCOBOL | Object Oriented COBOL |
| ORB | Object Request Broker |
| OS | Operating System |
| OSI | Open Systems Interconnection |
| OTS | Object Transaction Service |
| PC | Personal Computer |
| PDM II E2 | Practical Data Manager II Extended Version 2 |
| PIC | Plug-in Code |
| PNM | Public Network Management |
| POSIX | Portable Operating System Interface for UNIX |
| PP | Program Product |
| PR | Protected Retrieve |
| PU | Protected Update |
| RAID | Redundant Arrays of Inexpensive Disk |
| RD | Relational Database |
| RDB | Relational Database |
| RDB1 | Relational Database Manager 1 |
| RDB1 E2 | Relational Database Manager 1 Extended Version 2 |
| RDO | Remote Data Objects |
| RiSe | Real time SAN replication |
| RM | Resource Manager |
| RMM | Resource Manager Monitor |

| 英略語 | 英字の表記 |
|---|---|
| RPC | Remote Procedure Call |
| SAX | Simple API for XML |
| SDS | Single Database Server |
| SGML | Standard Generalized Markup Language |
| SJIS | Shift JIS |
| SNMP | Simple Network Management Protocol |
| SNTP | Simple Network Time Protocol |
| SQL | Structured Query Language |
| SQL/K | Structured Query Language / VOS K |
| SR | Shared Retrieve |
| SU | Shared Update |
| TCP/IP | Transmission Control Protocol / Internet Protocol |
| TM | Transaction Manager |
| TMS-4V/SP | Transaction Management System - 4V / System Product |
| UAP | User Application Program |
| UOC | User Own Coding |
| VOS1 | Virtual-storage Operating System 1 |
| VOS3 | Virtual-storage Operating System 3 |
| VOS K | Virtual-storage Operating System Kindness |
| WS | Workstation |
| WWW | World Wide Web |
| XDM/BASE E2 | Extensible Data Manager / Base Extended Version 2 |
| XDM/DF | Extensible Data Manager / Distributing Facility |
| XDM/DS | Extensible Data Manager / Data Spreader |
| XDM/RD E2 | Extensible Data Manager / Relational Database Extended Version 2 |
| XDM/SD E2 | Extensible Data Manager / Structured Database Extended Version 2 |
| XDM/XT | Extensible Data Manager / Data Extract |
| XFIT | Extended File Transmission program |
| XML | Extensible Markup Language |

## ■ログの表記

Windows のイベントビューアで表示されるアプリケーションログをイベントログと表記します。イベントログは，次の方法で参照できます。

〈手順〉

1.［スタート］－［プログラム］－［管理ツール（共通）］－［イベントビューア］を選択します。
2.［ログ］－［アプリケーション］を選択します。

アプリケーションログが表示されます。「ソース」の列が「HiRDBSingleServer」または「HiRDBParallelServer」になっているのが HiRDB が出力したメッセージです。

なお，セットアップ識別子を指定してインストールした場合は，「HiRDBSingleServer」または「HiRDBParallelServer」にセットアップ識別子が付いた名称となります。

## ■ Windows の操作説明で使う表記

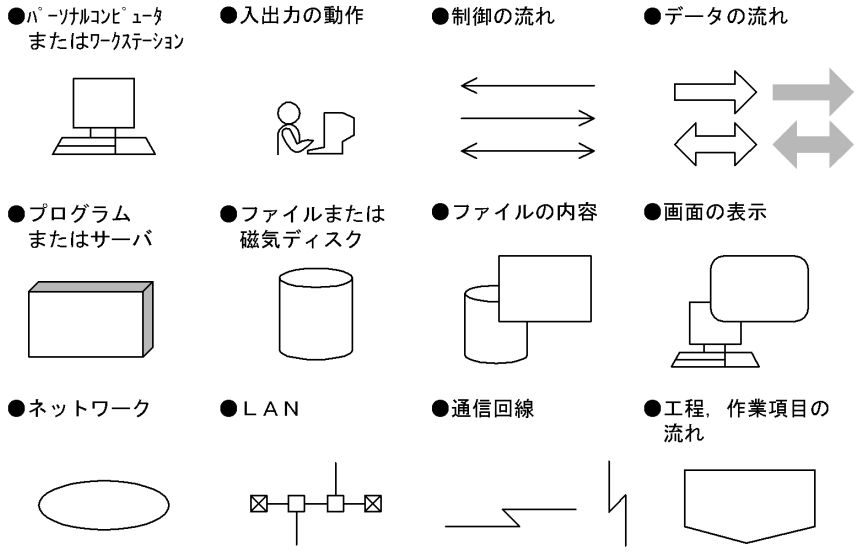
Windows の操作説明で使う記号を次に示します。

| 記　号 | 意　味 |
|---|---|
| ［ ］ | ボタンやテキストボックスなど，画面に表示されている要素を示します。 |
| ［ ］－［ ］ | 画面に表示されるメニューやアイコンなどを選択する操作を示します。 |

Windows の用語「ディレクトリ」と「フォルダ」は，「ディレクトリ」に統一して表記しています。

## ■図中で使用する記号

このマニュアルの図中で使用する記号を次のように定義します。




## ■このマニュアルで使用する計算式の記号

このマニュアルで使用する計算式の記号の意味を次に示します。

| 記号 | 意味 |
|---|---|
| ↑↑ | 計算結果の値を小数点以下で切り上げることを示します。<br>（例）↑ 34 ÷ 3 ↑の計算結果は 12 となります。 |
| ↓↓ | 計算結果の値を小数点以下で切り下げることを示します。<br>（例）↓ 34 ÷ 3 ↓の計算結果は 11 となります。 |
| MAX | 計算結果の最も大きい値を選ぶことを示しています。<br>（例）MAX（３×６，４＋７）の計算結果は 18 となります。 |
| MIN | 計算結果の最も小さい値を選ぶことを示しています。<br>（例）MIN（３×６，４＋７）の計算結果は 11 となります。 |
| mod | mod（ａ，ｂ）は，ａをｂで割った余りを示しています。<br>（例）mod（９，２）の計算結果は１となります。 |

## ■ Windows のパス名に関する注意

- パス名を絶対パスで指定する場合はドライブ名を指定してください。

（例）C:¥win32app¥hitachi¥hirdb_s¥spool¥tmp

- コマンドの引数，制御文ファイル，および HiRDB システム定義ファイル中に空白または丸括弧を含むパス名を指定する場合は，前後を引用符（ ”）で囲んでください。

（例）pdinit -d "C:¥Program Files(x86)¥hitachi¥hirdb_s¥conf¥mkinit"

ただし，バッチファイルもしくはコマンドプロンプト上で set コマンドを使用して環境変数を設定する場合，またはインストールディレクトリを指定する場合は引用符は不要です。引用符で囲むと，引用符も環境変数の値に含まれます。

（例）set PDCLTPATH=C:¥Program Files¥hitachi¥hirdb_s¥spool

- HiRDB はネットワークドライブのファイルを使用できないため，HiRDB のインストール，および環境構築はローカルドライブで行ってください。また，ユティリティの入出力ファイルなども，ローカルドライブ上のファイルを使用してください。
- パス名には，ショートパス名（例えば，C:¥PROGRA~1 など）は使用しないでください。

## ■ KB（キロバイト）などの単位表記について

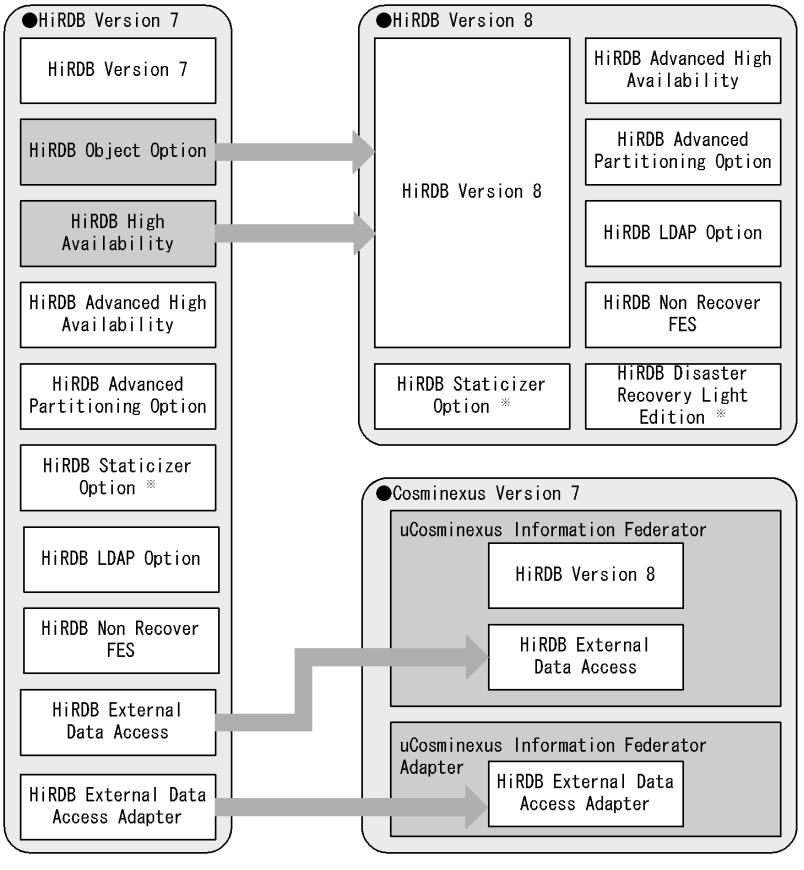
1KB（キロバイト），1MB（メガバイト），1GB（ギガバイト），1TB（テラバイト）はそれぞれ 1,024 バイト，$1{,}024^2$ バイト，$1{,}024^3$ バイト，$1{,}024^4$ バイトです。

## ■ Version 7 と Version 8 の製品体系の違い

HiRDB Version 8 では，HiRDB Version 7 までオプション製品（HiRDB Object Option および HiRDB High Availability）で提供していた機能を HiRDB の標準機能としました。それに伴い，オプション製品が廃止になりました。

また，Version 8 以降，HiRDB External Data Access および HiRDB External Data Access Adapter は HiRDB シリーズではなく，Cosminexus Version 7 シリーズとなりました。

HiRDB Version 7 と Version 8 の製品体系の違いを次に示します。




注※　UNIX版でだけ使用できる製品です。

# 目次


1 HiRDB のシステム構築の概要 1

1.1 システム構築手順 2
1.1.1 HiRDB を新規導入するときのシステム構築手順 2
1.1.2 HiRDB の環境設定の概要 2
1.1.3 ほかの製品と連携する場合の環境設定 3
1.1.4 Windows Terminal Service の使用 4
1.2 HiRDB のディレクトリ及びファイル構成 6
1.2.1 最初に作成するファイル 6
1.2.2 単調増加ファイル 9
1.3 HiRDB のバージョンアップ 28
1.3.1 バージョンアップ前にすること 28
1.3.2 旧バージョンと新バージョンを入れ替える場合 32
1.3.3 HiRDB のプラグインをバージョンアップする場合 34
1.3.4 Java ストアドプロシジャ及び Java ストアドファンクションを使用する場合 34
1.3.5 バージョンアップに失敗した場合 34
1.3.6 HiRDB を旧バージョンに戻す場合 36
1.4 修正版 HiRDB への入れ替え 38
1.4.1 修正版 HiRDB への入れ替え方法 38
1.4.2 前提条件 39
1.4.3 入れ替え前にすること 40
1.4.4 入れ替えの手順 40
1.4.5 系切り替え機能使用時の入れ替え手順 42
1.4.6 注意事項 43
1.4.7 運用上の注意事項 43
1.4.8 関連製品の制限及び注意事項 44
1.4.9 障害時の運用 44

2 インストール 47

2.1 インストール前に必要な作業 48
2.1.1 サーバマシン環境の確認 48
2.1.2 HiRDB 管理者の登録 51
2.1.3 OS 環境ファイルの設定 52
2.1.4 ホスト名の登録 52
2.2 HiRDB のインストール手順 54
2.2.1 インストール前の注意 54
2.2.2 HiRDB のインストール手順 54
2.2.3 HiRDB の環境変数 55
2.2.4 インストール後の注意 56
2.2.5 インストール時に 1073 エラーになったときの対処 57

2.3 インストール後の作業 58
2.3.1 文字コードの選択 58
2.3.2 HiRDB 運用ディレクトリ下のファイルの削除 58
2.3.3 ワークファイル出力先ディレクトリの作成 59
2.3.4 HiRDB ファイルシステム領域を作成する準備 60
2.3.5 Windows ファイアウォールの設定を有効にしている場合 61
2.4 付加 PP のインストール時の注意 62
2.5 HiRDB のアンインストール手順 63
2.6 Windows ファイアウォールの例外リストへの登録 66
2.6.1 HiRDB サーバのプログラムを例外リストに登録する 66
2.6.2 リモートシェル実行時に使用するポート番号を例外リストに登録する 67
2.6.3 UAP を例外リストに登録する 67
2.6.4 例外リストへの登録が不要になるケース 68
2.7 Windows ファイアウォールの例外リストからの削除 69
2.7.1 HiRDB サーバのプログラムを例外リストから削除する 69
2.7.2 リモートシェル実行時に使用するポート番号を例外リストから削除する 69
2.7.3 UAP を例外リストから削除する 70

3 簡易セットアップツールによる環境設定 71
3.1 簡易セットアップツールの概要 72
3.1.1 簡易セットアップツールとは 72
3.1.2 環境設定 72
3.1.3 定義の更新 76
3.1.4 簡易セットアップツールの稼働環境 77
3.1.5 簡易セットアップツールを実行する前に確認すること 78
3.2 簡易セットアップツールの起動 80
3.3 標準セットアップ 82
3.3.1 標準セットアップの場合の環境設定手順 82
3.3.2 標準セットアップで設定される環境 84
3.3.3 簡易セットアップツールで作成するサンプルデータベース 88
3.4 ウィザードセットアップ 90
3.4.1 ウィザードセットアップの場合の環境設定手順 90
3.5 カスタムセットアップ 95
3.5.1 カスタムセットアップの場合の環境設定手順 95
3.6 カスタムセットアップ（詳細定義） 97
3.6.1 メニュー一覧 97
3.6.2 カスタムセットアップ（詳細定義）の場合の環境設定手順 101
3.7 詳細定義情報の読み込み 120
3.8 定義の更新 121
3.9 HiRDB システム定義を編集する場合の注意事項 122
3.10 HiRDB/ パラレルサーバの環境設定をする場合 124

3.11 系切り替え機能の設定 128

4 コマンドによる環境設定 133

4.1 コマンドによる環境設定の概要 134
4.2 HiRDB システム定義の作成 136
4.2.1 HiRDB システム定義の作成（HiRDB/ シングルサーバの場合） 136
4.2.2 HiRDB システム定義の作成（HiRDB/ パラレルサーバの場合） 138
4.2.3 HiRDB システム定義（UAP 環境定義を除く）の変更方法 144
4.2.4 UAP 環境定義の追加又は変更方法 146
4.3 HiRDB ファイルシステム領域の作成 147
4.3.1 HiRDB ファイルシステム領域の種類 147
4.3.2 raw I/O 機能を使用した HiRDB ファイルシステム領域の作成 147
4.3.3 例題 1（RD エリア用の HiRDB ファイルシステム領域の作成） 149
4.3.4 例題 2（システムファイル用の HiRDB ファイルシステム領域の作成） 150
4.3.5 例題 3（作業表用ファイル用の HiRDB ファイルシステム領域の作成） 150
4.3.6 例題 4（ユティリティ用の HiRDB ファイルシステム領域の作成） 151
4.3.7 例題 5（リスト用 RD エリア用の HiRDB ファイルシステム領域の作成） 151
4.3.8 例題 6（raw I/O 機能を使用する HiRDB ファイルシステム領域の作成） 152
4.4 システムファイルの作成 153
4.4.1 システムログファイルの作成 153
4.4.2 シンクポイントダンプファイルの作成 153
4.4.3 ステータスファイルの作成 154
4.4.4 システムファイルの作成例（HiRDB/ シングルサーバの場合） 155
4.4.5 システムファイルの作成例（HiRDB/ パラレルサーバの場合） 157
4.5 システム用 RD エリアの作成 165
4.5.1 基本事項 165
4.5.2 例題 1（HiRDB/ シングルサーバの場合） 165
4.5.3 例題 2（HiRDB/ パラレルサーバの場合） 167
4.6 HiRDB の初期開始 169
4.7 ユーザ用 RD エリアの作成 170
4.7.1 基本事項 170
4.7.2 例題 1（HiRDB/ シングルサーバの場合） 170
4.7.3 例題 2（HiRDB/ パラレルサーバの場合） 171
4.8 ユーザ LOB 用 RD エリアの作成 173
4.8.1 基本事項 173
4.8.2 例題 1（HiRDB/ シングルサーバの場合） 173
4.8.3 例題 2（HiRDB/ パラレルサーバの場合） 175
4.9 データディクショナリ LOB 用 RD エリアの作成 177
4.9.1 基本事項 177
4.9.2 例題 1（HiRDB/ シングルサーバの場合） 177
4.9.3 例題 2（HiRDB/ パラレルサーバの場合） 178

4.10 リスト用 RD エリアの作成 181
4.10.1 基本事項 181
4.10.2 例題 1（HiRDB/ シングルサーバの場合） 181
4.10.3 例題 2（HiRDB/ パラレルサーバの場合） 182

5 プラグインの環境設定 185
5.1 プラグインの環境設定の概要 186
5.1.1 環境設定手順 186
5.1.2 プラグイン使用時の注意 190
5.2 プラグインのバージョンアップ 192
5.2.1 バージョンアップ時の注意 192
5.2.2 バージョンアップの手順 192
5.3 プラグインの削除 194

6 データベースの作成 197
6.1 データベース作成の概要 198
6.1.1 データベースを作成する前に必要な作業 198
6.1.2 データベースの作成手順 199
6.1.3 データベースの更新ログ取得方式 199
6.1.4 ユニーク属性のインデクスを定義した表にデータロードする場合の注意 202
6.1.5 大量のデータをロードする場合（同期点指定のデータロード） 202
6.1.6 横分割表にデータをロードする場合（分割入力データファイルの作成） 203
6.1.7 自動採番機能を使用したデータロード 203
6.1.8 入力データファイル UOC 205
6.1.9 不要な RD エリアの削除 205
6.2 横分割表の作成 206
6.3 LOB 列を定義した表の作成 209
6.4 プラグインが提供する抽象データ型を定義した表の作成 212
6.4.1 SGMLTEXT 型 212
6.4.2 XML 型 215
6.5 ユーザが定義した抽象データ型を定義した表の作成 230
6.5.1 抽象データ型の定義 230
6.5.2 表の定義 233
6.5.3 インデクスの定義 234
6.5.4 表へのデータの格納 234
6.5.5 データベースの更新ログ取得方式 235
6.5.6 データの格納状態の確認 236
6.6 インデクス一括作成中に発生したエラーの対処方法 238
6.6.1 ログ取得モード又は更新前ログ取得モードでデータロードをしていた場合 238
6.6.2 ログレスモードでデータロードをしていた場合 240
6.7 同期点指定のデータロード実行中にユティリティが異常終了したときの対処方法 243

6.7.1 対処方法の概要 243
6.7.2 例題 244

7 ほかの製品との連携 247
7.1 レプリケーション機能との連携 248
7.1.1 HiRDB Datareplicator との連携 248
7.1.2 HiRDB Dataextractor との連携 248
7.2 OLTP との連携 249
7.2.1 OLTP と連携できる製品 249
7.2.2 HiRDB XA ライブラリ 249
7.2.3 OLTP と連携した HiRDB システムの構成例 251
7.2.4 トランザクションの移行 253
7.2.5 トランザクションマネジャへの登録 255
7.2.6 トランザクションマネジャに登録する情報 256
7.2.7 トランザクションマネジャへの登録例 260
7.2.8 トランザクションマネジャへの登録の変更 262
7.2.9 トランザクションマネジャと HiRDB 間のコネクションが切断されたときの再接続方法 263
7.2.10 注意事項 264
7.3 HiRDB External Data Access 機能との連携 266
7.4 JP1 との連携 267
7.4.1 JP1/Base へのイベント通知 267
7.4.2 JP1/IM によるイベント管理 269
7.4.3 JP1/AJS2 との連携によるジョブの自動実行 270

8 HiRDB/ シングルサーバの設計 271
8.1 HiRDB/ シングルサーバのシステム設計 272
8.1.1 システム設計 272
8.1.2 HiRDB/ シングルサーバのシステム構成 273
8.2 HiRDB ファイルシステム領域の設計 275
8.2.1 RD エリア用の HiRDB ファイルシステム領域の設計 275
8.2.2 システムファイル用の HiRDB ファイルシステム領域の設計 275
8.2.3 作業表用ファイル用の HiRDB ファイルシステム領域の設計 276
8.2.4 ユティリティ用の HiRDB ファイルシステム領域の設計 276
8.2.5 リスト用 RD エリア用の HiRDB ファイルシステム領域の設計 277
8.2.6 HiRDB ファイルシステム領域の最大長 277
8.2.7 系切り替え機能使用時の注意事項 278
8.3 システムファイルの設計 279
8.3.1 システムログファイルの設計 279
8.3.2 シンクポイントダンプファイルの設計 282
8.3.3 ステータスファイルの設計 284
8.4 RD エリアの配置 288

8.4.1 システム用 RD エリアの配置 288
8.4.2 データディクショナリ LOB 用 RD エリアの配置 288
8.4.3 ユーザ用 RD エリアの配置 289
8.4.4 ユーザ LOB 用 RD エリアの配置 290
8.4.5 リスト用 RD エリアの配置 290

9 HiRDB/ パラレルサーバの設計 291
9.1 HiRDB/ パラレルサーバのシステム設計 292
9.1.1 システム設計 292
9.1.2 HiRDB/ パラレルサーバのシステム構成 294
9.1.3 マルチフロントエンドサーバの設定 295
9.1.4 回復不要 FES 298
9.2 HiRDB ファイルシステム領域の設計 303
9.2.1 RD エリア用の HiRDB ファイルシステム領域の設計 303
9.2.2 システムファイル用の HiRDB ファイルシステム領域の設計 304
9.2.3 作業表用ファイル用の HiRDB ファイルシステム領域の設計 304
9.2.4 ユティリティ用の HiRDB ファイルシステム領域の設計 305
9.2.5 リスト用 RD エリア用の HiRDB ファイルシステム領域の設計 306
9.2.6 HiRDB ファイルシステム領域の最大長 306
9.2.7 系切り替え機能使用時の注意事項 306
9.3 システムファイルの設計 307
9.3.1 システムログファイルの設計 307
9.3.2 シンクポイントダンプファイルの設計 310
9.3.3 ステータスファイルの設計 312
9.4 RD エリアの配置 316
9.4.1 システム用 RD エリアの配置 316
9.4.2 データディクショナリ LOB 用 RD エリアの配置 317
9.4.3 ユーザ用 RD エリアの配置 317
9.4.4 ユーザ LOB 用 RD エリアの配置 318
9.4.5 リスト用 RD エリアの配置 319
9.5 大規模なシステムの構築 320
9.6 HiRDB/ パラレルサーバのヘテロ構成 322

10 マルチ HiRDB の設計 323
10.1 マルチ HiRDB のシステム設計 324
10.1.1 マルチ HiRDB のインストール 324
10.1.2 マルチ HiRDB の環境設定 325
10.2 運用上の注意 327

11 グローバルバッファ，ローカルバッファの設計 329
11.1 グローバルバッファの割り当て 330
11.1.1 インデクス用グローバルバッファの割り当て 330
11.1.2 データ用グローバルバッファの割り当て 331
11.1.3 LOB 用グローバルバッファの割り当て 332
11.1.4 グローバルバッファの割り当て方法 333
11.2 グローバルバッファのバッファ面数の設定 335
11.3 プリフェッチ機能の指定 336
11.4 非同期 READ 機能の指定 337
11.5 デファードライト処理の指定 338
11.6 デファードライト処理の並列 WRITE 機能の指定 339
11.7 コミット時反映処理の設定 340
11.8 グローバルバッファの LRU 管理方式 341
11.8.1 LRU の管理方式 341
11.8.2 UAP ごとの LRU 管理抑止設定 342
11.8.3 UAP がアクセスするバイナリデータの LRU 管理抑止設定 343
11.9 スナップショット方式によるページアクセス 346
11.10 グローバルバッファの先読み入力 348
11.11 ローカルバッファ 350
11.11.1 インデクス用ローカルバッファの割り当て 350
11.11.2 データ用ローカルバッファの割り当て 350
11.11.3 ローカルバッファの割り当て方法 351
11.11.4 ローカルバッファ使用時の注意 351

12 表の設計 353
12.1 表を設計するときの検討項目 354
12.2 表の正規化 357
12.3 表の横分割 361
12.3.1 表の横分割の概要 361
12.3.2 表の横分割の種類 361
12.3.3 表の横分割の形態 369
12.3.4 表の横分割の効果 370
12.3.5 設計上の考慮点 371
12.3.6 表を横分割する場合の注意 377
12.4 表のマトリクス分割 378
12.5 トリガの定義 382
12.5.1 適用基準 382
12.5.2 トリガの定義 383
12.5.3 トリガの使用上の注意 386
12.5.4 トリガの管理 386

12.5.5 障害時の回復方法 389
12.6 ビュー表の作成 391
12.7 FIX 属性の指定 393
12.8 主キー（プライマリキー）の指定 394
12.9 クラスタキーの指定 395
12.10 サプレスオプションの指定 397
12.11 ノースプリットオプションの指定 398
12.12 バイナリデータ列の指定 400
12.12.1 BLOB 型 401
12.12.2 BINARY 型 401
12.12.3 BLOB 型と BINARY 型の使い分け 402
12.13 文字集合の指定 404
12.14 WITHOUT ROLLBACK オプションの指定 405
12.15 改竄防止機能の指定 407
12.15.1 指定方法 407
12.15.2 制限事項 408
12.15.3 非改竄防止表から改竄防止表への変更 411
12.15.4 障害時の運用 417
12.16 繰返し列を含む表 418
12.17 抽象データ型を含む表 420
12.18 共用表 424
12.18.1 効果と適用基準 425
12.18.2 定義方法 426
12.18.3 共用表の操作 426
12.18.4 共用表の制限事項 427
12.18.5 定義系 SQL，ユティリティ，及び運用コマンド実行時の注意 428
12.18.6 HiRDB/ シングルサーバで共用表を使用する場合 428
12.19 参照制約 430
12.19.1 参照制約とは 430
12.19.2 参照制約の定義 430
12.19.3 検査保留状態 438
12.19.4 データ操作と整合性 443
12.19.5 表の整合性確認手順 444
12.19.6 参照制約とトリガ 449
12.19.7 関連製品との連携時の注意 452
12.20 検査制約 453
12.20.1 検査制約とは 453
12.20.2 検査制約の定義 453
12.20.3 検査保留状態 454
12.20.4 データ操作と整合性 455
12.20.5 表の整合性確認手順 456
12.20.6 関連製品との連携時の注意 459

13 インデクスの設計 461
13.1 インデクスを設計するときの検討項目 462
13.2 インデクス 463
13.2.1 インデクスの作成 463
13.2.2 コストベースの最適化を考慮したインデクスの作成 464
13.2.3 単一列インデクスと複数列インデクスの使い分け 467
13.2.4 複数のインデクスの使用 469
13.2.5 除外キー値を設定したインデクスの使用 469
13.2.6 インデクスの数が性能に与える影響 469
13.3 インデクスの横分割 471
13.4 プラグインインデクス 476
13.5 プラグインインデクスの横分割 477

14 RD エリアの設計 483
14.1 RD エリアを設計するときの検討項目 484
14.2 セグメント 487
14.2.1 セグメントサイズの決定 487
14.2.2 セグメント内の空きページ比率の設定 488
14.2.3 セグメントの確保と解放 489
14.3 ページ 490
14.3.1 ページ長の決定 490
14.3.2 ページ内の未使用領域の比率の設定 491
14.3.3 ページの確保と解放 493
14.4 リスト用 RD エリアの設計 494
14.5 空き領域の再利用機能 496
14.5.1 データ格納時のサーチ方式 496
14.5.2 空き領域の再利用機能とは 496
14.5.3 効果と適用基準 498
14.5.4 使用前の考慮点 498
14.5.5 環境設定 499
14.5.6 実行状態の確認 500
14.5.7 注意事項 501
14.6 共用 RD エリア（HiRDB/ パラレルサーバ限定） 502

15 HiRDB のメモリ所要量 505
15.1 HiRDB/ シングルサーバのメモリ所要量の見積もり 506
15.1.1 メモリ配置 506
15.1.2 メモリ所要量の計算式 508
15.1.3 ユニットコントローラが使用する共用メモリの計算式 515

15.1.4 シングルサーバが使用する共用メモリの計算式 523
15.1.5 グローバルバッファが使用する共用メモリの計算式 527
15.1.6 SQL 実行時に必要なメモリ所要量の計算式 529
15.1.7 SQL 前処理時に必要なメモリ所要量の計算式 536
15.1.8 BLOB 型データの検索又は更新時に必要なメモリ所要量の計算式（HiRDB/ シングルサーバの場合） 537
15.1.9 ブロック転送又は配列 FETCH で必要なメモリ所要量の計算式 538
15.1.10 インメモリデータ処理で必要なメモリ所要量 539
15.2 HiRDB/ パラレルサーバのメモリ所要量の見積もり 540
15.2.1 メモリ配置 540
15.2.2 メモリ所要量の計算式 542
15.2.3 ユニットコントローラが使用する共用メモリの計算式 550
15.2.4 各サーバが使用する共用メモリの計算式 572
15.2.5 グローバルバッファが使用する共用メモリの計算式 579
15.2.6 SQL 実行時に必要なメモリ所要量の計算式 585
15.2.7 SQL 前処理時に必要なメモリ所要量の計算式 591
15.2.8 BLOB 型データの検索又は更新時に必要なメモリ所要量の計算式（フロントエンドサーバの場合） 593
15.2.9 BLOB 型データの検索又は更新時に必要なメモリ所要量の計算式（バックエンドサーバ又はディクショナリサーバの場合） 594
15.2.10 ブロック転送又は配列 FETCH で必要なメモリ所要量の計算式（フロントエンドサーバの場合） 594
15.2.11 インメモリデータ処理で必要なメモリ所要量 595

16 RD エリアの容量の見積もり 597
16.1 ユーザ用 RD エリアの容量の見積もり 598
16.1.1 ユーザ用 RD エリアの容量の計算方法 598
16.1.2 表の格納ページ数の計算方法 599
16.1.3 インデクスの格納ページ数の計算方法 609
16.2 データディクショナリ用 RD エリアの容量の見積もり 618
16.2.1 通常のデータディクショナリ用 RD エリアの容量の見積もり 618
16.2.2 解析情報表及び運用履歴表を格納するデータディクショナリ用 RD エリアの容量の見積もり 649
16.3 マスタディレクトリ用 RD エリアの容量の見積もり 652
16.4 データディレクトリ用 RD エリアの容量の見積もり 653
16.5 データディクショナリ LOB 用 RD エリアの容量の見積もり 654
16.6 ユーザ LOB 用 RD エリアの容量の見積もり 660
16.7 レジストリ用 RD エリアの容量の見積もり 661
16.8 レジストリ LOB 用 RD エリアの容量の見積もり 663
16.9 リスト用 RD エリアの容量の見積もり 664

17 システムファイル及び監査証跡ファイルの容量の見積もり 665
17.1 システムログファイルの容量の見積もり 666
17.1.1 システムログファイルの総容量 666
17.1.2 表定義時に出力されるシステムログ量 668

17.1.3 インデクス定義時に出力されるシステムログ量 669
17.1.4 表データ更新時に出力されるシステムログ量 671
17.1.5 ユティリティによるデータベース作成時に出力されるシステムログ量 680
17.1.6 SQL 操作に応じて出力されるシステムログ量 682
17.1.7 RD エリアの自動増分機能使用時に出力されるシステムログ量 682
17.1.8 PURGE TABLE 文実行時に出力されるシステムログ量 682
17.1.9 空きページ解放ユティリティ（pdreclaim）実行時に出力されるシステムログ量 683
17.1.10 再編成時期予測機能使用時に出力されるシステムログ量 684
17.2 シンクポイントダンプファイルの容量の見積もり 686
17.3 ステータスファイルの容量の見積もり 687
17.4 監査証跡ファイルの容量の見積もり 693

18 作業表用ファイルの容量の見積もり 695
18.1 作業表用ファイルの容量の見積もりの概要 696
18.2 HiRDB ファイルシステム領域サイズの見積もり（pdfmkfs -n コマンド） 697
18.2.1 SQL 文が使用する作業表用ファイルの容量 697
18.2.2 ユティリティが使用する作業表用ファイルの容量 701
18.3 最大ファイル数の見積もり（pdfmkfs -l コマンド） 704
18.4 最大増分回数の見積もり（pdfmkfs -e コマンド） 706

19 ユティリティ実行時の容量の見積もり 707
19.1 ユティリティ実行時のファイルの容量の見積もり 708
19.1.1 データベース作成ユティリティ（pdload）実行時のファイルの容量 708
19.1.2 データベース再編成ユティリティ（pdrorg）実行時のファイルの容量 710
19.1.3 統計解析ユティリティ（pdstedit）実行時のファイルの容量 714
19.1.4 データベース状態解析ユティリティ（pddbst）実行時のファイルの容量 716
19.1.5 データベース複写ユティリティ（pdcopy）実行時のファイルの容量 716
19.1.6 ディクショナリ搬出入ユティリティ（pdexp）実行時のファイルの容量 719
19.1.7 最適化情報収集ユティリティ（pdgetcst）実行時のファイルの容量 720
19.1.8 アクセスパス表示ユティリティ（pdvwopt）実行時のファイルの容量 721
19.1.9 リバランスユティリティ（pdrbal）実行時のファイルの容量 721
19.1.10 整合性チェックユティリティ（pdconstck）実行時のファイルの容量 723
19.2 ユティリティ実行時のメモリ所要量の見積もり 724
19.2.1 データベース初期設定ユティリティ（pdinit）実行時のメモリ所要量 724
19.2.2 データベース定義ユティリティ（pddef）実行時のメモリ所要量 725
19.2.3 データベース作成ユティリティ（pdload）実行時のメモリ所要量 725
19.2.4 データベース再編成ユティリティ（pdrorg）実行時のメモリ所要量 728
19.2.5 データベース構成変更ユティリティ（pdmod）実行時のメモリ所要量 730
19.2.6 統計解析ユティリティ（pdstedit）実行時のメモリ所要量 732
19.2.7 データベース状態解析ユティリティ（pddbst）実行時のメモリ所要量 732
19.2.8 最適化情報収集ユティリティ（pdgetcst）実行時のメモリ所要量 733

19.2.9 データベース複写ユティリティ（pdcopy）実行時のメモリ所要量 734
19.2.10 データベース回復ユティリティ（pdrstr）実行時のメモリ所要量 735
19.2.11 ディクショナリ搬出入ユティリティ（pdexp）実行時のメモリ所要量 737
19.2.12 アクセスパス表示ユティリティ（pdvwopt）実行時のメモリ所要量 738
19.2.13 リバランスユティリティ（pdrbal）実行時のメモリ所要量 739
19.2.14 空きページ解放ユティリティ（pdreclaim）及びグローバルバッファ常駐化ユティリティ（pdpgbfon）実行時のメモリ所要量 741
19.2.15 整合性チェックユティリティ（pdconstck）実行時のメモリ所要量 742

20 リソース数に関連する環境変数の見積もり 745
20.1 HiRDB/ シングルサーバの場合 746
20.2 HiRDB/ パラレルサーバの場合 749

21 Windows レジストリ設定値の見積もり 753
21.1 デスクトップヒープ指定値の見積もり 754

22 サンプルファイル 757
22.1 サンプルファイルの概要 758
22.1.1 サンプルファイルのファイル名 758
22.2 表の定義情報 761
22.3 サンプルファイルの使用方法 763
22.3.1 サンプルデータベースの作成手順 763
22.3.2 サンプルデータベースのカスタマイズ 763
22.3.3 サンプルで使用する HiRDB ファイルシステム領域名とユーザ作成ファイル名 766

23 HiRDB サーバと HiRDB クライアント間の通信 769
23.1 HiRDB サーバと HiRDB クライアントの接続方法 770
23.1.1 FQDN を指定した HiRDB サーバへの接続方法 770
23.1.2 マルチコネクションアドレス機能を使用した HiRDB サーバへの接続方法 771
23.2 DNS サーバで IP アドレスを管理する場合の設定 776
23.3 ファイアウォールや NAT が設置されている場合の設定 778
23.3.1 HiRDB/ シングルサーバ側にファイアウォールを設置した場合 778
23.3.2 HiRDB/ シングルサーバ側にファイアウォールと NAT を設置した場合 779
23.3.3 HiRDB/ パラレルサーバ側にファイアウォールを設置した場合 780
23.3.4 HiRDB/ パラレルサーバ側にファイアウォールと NAT を設置した場合 781
23.4 HiRDB が使用するポート数 783
23.4.1 ユニットが使用する通信ポート数の見積もり 783
23.4.2 注意事項 783
23.4.3 計算例 784

23.5 HiRDB で指定するポート番号 785
23.5.1 HiRDB で指定するポート番号の一覧 785
23.5.2 ポート番号の指定方法 785
23.5.3 ポート番号の重複に関する注意事項 788
23.6 HiRDB 予約ポート機能 789
23.6.1 HiRDB 予約ポート数の見積もり 789

付録 791
付録 A HiRDB の最大値・最小値 792
付録 A.1 システム構成に関する最大値と最小値 792
付録 A.2 データベースに関する最大値と最小値 793
付録 A.3 HiRDB ファイル名に関する最大値と最小値 794
付録 B HiRDB のプロセス一覧 796
付録 B.1 HiRDB/ シングルサーバで起動するプロセス 796
付録 B.2 HiRDB/ パラレルサーバで起動するプロセス 800
付録 C Q & A 808
付録 D バッチファイルによる環境設定 817
付録 D.1 バッチファイルによる環境設定の概要 817
付録 D.2 バッチファイルによる HiRDB の環境設定 822
付録 D.3 HiRDB システム定義の変更 826

索引 827

# 1 HiRDB のシステム構築の概要

この章では，HiRDB のシステム構築手順，HiRDB のファイル構成及びバージョンアップ手順について説明します。


1.1 システム構築手順

1.2 HiRDB のディレクトリ及びファイル構成

1.3 HiRDB のバージョンアップ

1.4 修正版 HiRDB への入れ替え

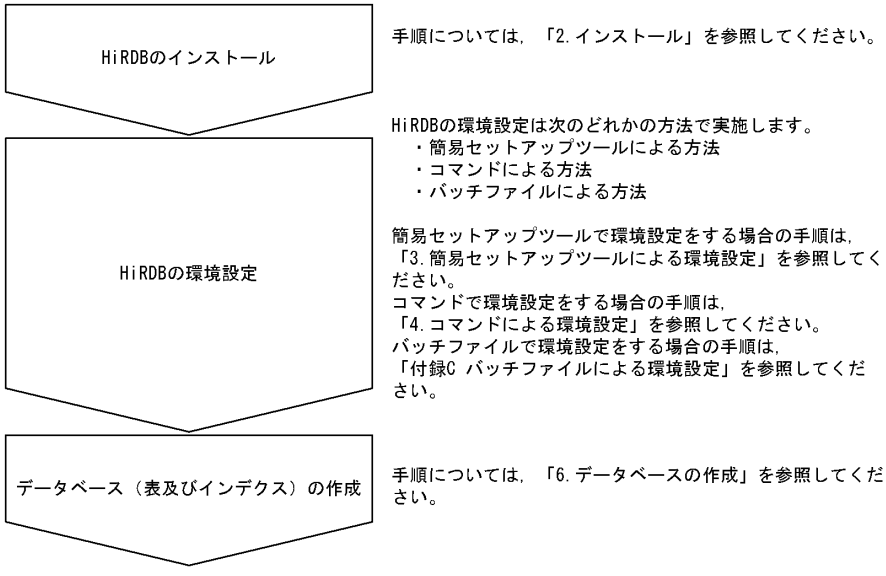
# 1.1 システム構築手順

ここでは，HiRDB を新規導入するときのシステム構築手順について説明します。

## 1.1.1 HiRDB を新規導入するときのシステム構築手順

HiRDB を新規導入するときのシステム構築手順を次の図に示します。

**図 1-1 HiRDB を新規導入するときのシステム構築手順**




なお，HiRDB を 24 時間連続稼働するときにお勧めする運用方法，及び注意事項について，マニュアル「HiRDB Version 8 システム運用ガイド」で説明しています。必要に応じて参照してください。

## 1.1.2 HiRDB の環境設定の概要

HiRDB 管理者は，次に示すどれかの方法で HiRDB の環境設定をしてください。

- 簡易セットアップツールを使用する方法
- コマンドを使用する方法
- バッチファイル（SPsetup.bat）を使用する方法

**ポイント**

通常は，簡易セットアップツールを使用して HiRDB の環境設定をしてください。

各環境設定方法のメリット及びデメリットを次の表に示します。

表 1-1 各環境設定方法のメリット及びデメリット

| 環境設定方法 | 概要 | メリット | デメリット |
|---|---|---|---|
| 簡易セットアップツールを使用する方法 | HiRDB の環境設定情報を，表示される画面に従って入力していきます。その入力情報を基に HiRDB の環境設定が行われます。<br>簡易セットアップツールを使用する方法については，「3. 簡易セットアップツールによる環境設定」を参照してください。 | ほかの方法よりも手軽に HiRDB の環境設定ができます。簡易セットアップツールを使用して HiRDB の環境設定をすれば，取りあえず HiRDB を開始できるようになります。また，既存のシステム定義の指定値を変更することもできます。 | HiRDB のシステム構成が簡易セットアップツールで設定できる範囲に限定されます。 |
| コマンドを使用する方法※1 | HiRDB のコマンドを使用して HiRDB の環境設定を行います。コマンドを使用する方法については，「4. コマンドによる環境設定」を参照してください。 | HiRDB のシステム構成を自由に決められます。 | HiRDB の環境設定をするのに，ある一定以上の知識が必要になります。具体的には，このマニュアルで説明している機能や設定内容を理解しておく必要があります。そのため，ほかの方法よりも，環境設定方法が難しくなります。 |
| バッチファイル（SPsetup.bat）を使用する方法※2 | バッチファイルを実行して HiRDB の環境設定を行います。バッチファイルを使用する方法については，「付録 D バッチファイルによる環境設定」を参照してください。 | コマンドを使用する方法よりも手軽に HiRDB の環境設定ができます。バッチファイルを使用して HiRDB の環境設定をすれば，取りあえず HiRDB を開始できるようになります。 | HiRDB のシステム構成がバッチファイルで設定できる範囲に限定されます。 |

注※ 1

本番用のシステムを構築する前に簡易導入をお試しください。サンプルファイルを使ってテスト用のシステムで HiRDB の構築手順を一通り実行しておけば，本番用のシステムをより適切に構築できます。

簡易導入の方法については，「22. サンプルファイル」を参照してください。HiRDB/ シングルサーバの場合は，マニュアル「HiRDB ファーストステップガイド」を参照してください。

注※ 2

バッチファイルを使用する方法は，HiRDB/ シングルサーバのときだけ使えます。HiRDB/ パラレルサーバの環境設定については，%PDDIR%¥HiRDEF¥readme.txt を参照してください。

**注意事項**

- 簡易セットアップツールの場合，プラグインの環境設定はできません。
- バッチファイルの場合，HiRDB の環境設定情報を自動的に設定します。設定し終わった内容を基に，HiRDB 管理者が適切な環境に変更します。

## 1.1.3 ほかの製品と連携する場合の環境設定

ほかの製品と連携する場合の環境設定を次に示します。

### (1) レプリケーション機能を使用する場合

レプリケーション機能を使用するには，HiRDB Datareplicator，HiRDB Dataextractor が必要になります。レプリケーション機能の環境設定方法については，「7.1 レプリケーション機能との連携」を参照し

てください。

### (2) OLTP と連携する場合

OLTP と連携する場合の環境設定方法については，「7.2　OLTP との連携」を参照してください。

### (3) 系切り替え機能を使用する場合

系切り替え機能を使用する場合は，クラスタソフトウェアが必要になります。クラスタソフトウェアはプラットフォームごとに異なります。クラスタソフトウェア，及び系切り替え機能の環境設定方法については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

### (4) ディレクトリサーバ連携機能を使用する場合

HiRDB は次のディレクトリサーバと連携できます。

- Sun Java System Directory Server

ディレクトリサーバの環境設定方法については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

### (5) HiRDB External Data Access 機能を使用する場合

HiRDB External Data Access 機能を使用するには，HiRDB External Data Access が必要になります。HiRDB External Data Access 機能の環境設定方法については，「7.3　HiRDB External Data Access 機能との連携」を参照してください。

### (6) JP1 と連携する場合

JP1 と連携して，HiRDB のイベントを JP1 イベントとして管理できます。これによって，システムのイベントを一元管理したり，イベントを契機としてジョブを自動起動したりできます。JP1 との連携については，「7.4　JP1 との連携」を参照してください。

## 1.1.4　Windows Terminal Service の使用

Windows Terminal Service を使用して HiRDB の運用・操作ができます。これによって，遠隔地のマシンや，コンソールがないマシンに対しても HiRDB の運用・操作を行なうことが可能となります。Windows Terminal Service の詳細については，OS のマニュアルを参照してください。

Windows Terminal Service には，コンソールセッションと仮想セッションがあります。

**● コンソールセッション**

操作するサーバの入出力デバイス（ディスプレイ，キーボード，マウスなど）をターミナルサーバのクライアントのデバイスに切り替えるセッションです。主な注意事項を次に示します。

- サーバへリモートデスクトップ接続するコマンドに，/console オプションを付けて起動します。
- クライアント及びサーバの中で 1 個のセッションしか作成できません。
  （元の画面は操作不可の状態（ログオフ状態）となります）
- Windows 2000，Windows Vista，及び Windows Server 2008 ではサポートしていません。

**● 仮想セッション**

操作するサーバをバックグラウンドで操作する（クライアント操作用の画面をバックグラウンドに作成する）セッションです。主な注意事項を次に示します。

- サーバへリモートデスクトップで接続するコマンドをオプションなしで起動します。
- 複数作成可能です。

## (1) 注意事項

### (a) OS がインストールされているディレクトリ下のファイルアクセス

Windows Terminal Service を経由してアプリケーションを操作する場合，OS がインストールされているディレクトリ下（例えば c:¥windows）の初期化ファイル（ファイル拡張子が INI のファイル）ではなく，ユーザごとに作成されるディレクトリ下（例えば c:¥windows¥Document and Setting¥ ログインユーザ名¥Windows）の初期化ファイルが有効になるときがあります。

この場合，OS がインストールされているディレクトリ下の HiRDB.INI ファイルの内容を，ユーザごとに作成されるディレクトリ下の HiRDB.INI ファイルに設定する必要があります。ただし，運用コマンド，ユティリティ，クライアントプログラムを実行するコマンドプロンプトなどの実行環境に，クライアント環境定義を設定することで HiRDB.INI ファイルの場所の違いによる影響を回避できます。クライアント環境定義の設定方法については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

### (b) セキュリティ上の注意事項

Windows Terminal Service を使用した場合，サーバのコンソールがログインしていない状態であっても，クライアント側の画面がログインした状態の場合があります。このため，サーバ及びクライアントに次の設定又は運用を行うなどの注意が必要となります。

- スクリーンセーバーのパスワードによる保護を付ける
- クライアントからの操作が終了した場合，ログオフする

### (c) 制限事項

Windows Terminal Service 使用時，Windows Server 2003 (IPF) で系切り替え機能を使用する場合に制限があります。Windows Terminal Service 使用時の Windows Server 2003 (IPF) の系切り替え機能のサポート項目を次の表に示します。

**表 1-2 Windows Terminal Service 使用時の Windows Server 2003 (IPF) の系切り替え機能のサポート項目**

| 種別 | | | サポートの有無 |
|---|---|---|---|
| 系切り替え機能 | モニタモード | | ○ |
| | サーバモード※1 | コンソールセッション | ○ |
| | | 仮想セッション | ×※2 |

（凡例）
○：サポートしています。　×：サポートしていません。

注※1
ユニット制御情報定義に pd_ha_acttype = server を指定した場合です。

注※2
サーバ PC で操作，又はコンソールセッションで操作する必要があります。pdstart，pdstop，pdprgrenew，pdchgconf などのコマンドを操作した場合，Hitachi HA Toolkit Extension に HiRDB の状態を設定できないため，系切り替えが発生します。

# 1.2 HiRDB のディレクトリ及びファイル構成

## 1.2.1 最初に作成するファイル

### (1) HiRDB 管理者が作成するディレクトリ及びファイル

HiRDB 管理者が作成するディレクトリ及びファイル構成を次の表に示します。

**表 1-3 HiRDB 管理者が作成するディレクトリ及びファイル構成**

| ファイル又はディレクトリ名 | 説明 |
|---|---|
| %PDDIR%¥conf¥pdsys | システム共通定義を格納するファイル |
| %PDDIR%¥conf¥pdutsys | ユニット制御情報定義を格納するファイル |
| %PDDIR%¥conf¥pdsvrc | サーバ共通定義を格納するファイル |
| %PDDIR%¥conf¥ サーバ名 | 各サーバ定義を格納するファイル |
| %PDDIR%¥conf¥pduapenv | UAP 環境定義を格納するディレクトリ |
| %PDDIR%¥conf¥chgconf | システム構成変更用定義ファイルを格納するディレクトリ |

### (2) HiRDB が作成するディレクトリ及びファイル

HiRDB が作成するディレクトリ及びファイル構成を次の表に示します。

**表 1-4 HiRDB が作成するディレクトリ及びファイル構成**

| ファイル又はディレクトリ名 | 説明 |
|---|---|
| %PDDIR%¥bin | HiRDB のコマンド及びユティリティを格納するディレクトリ |
| %PDDIR%¥lib | HiRDB の共用ライブラリ，及びメッセージテキストファイルを格納するディレクトリ |
| %PDDIR%¥lib¥sysconf | HiRDB システム定義の定義解析用ファイルを格納するディレクトリ |
| %PDDIR%¥lib¥sysdef | |
| %PDDIR%¥lib¥servers | HiRDB サーバの実行ファイル及びライブラリを格納するディレクトリ |
| %PDDIR%¥client¥lib | HiRDB クライアントのライブラリを格納するディレクトリ |
| %PDDIR%¥client¥utl | HiRDB クライアントのコマンド及びユティリティを格納するディレクトリ |
| %PDDIR%¥client¥jba | HiRDB Java ストアドプロシジャ／ファンクション配布ウィザードを格納するディレクトリ（Windows 又は Windows（x64）の場合に作成されます） |
| %PDDIR%¥client¥include | UAP を作成するときに使用するヘッダ情報を格納するディレクトリ |
| %PDDIR%¥include | UAP を作成するときに使用するヘッダ情報を格納するディレクトリ |
| %PDDIR%¥spool¥tmp | 作業用一時ファイル格納ディレクトリ |
| %PDDIR%¥spool¥save ※1 | 退避コアファイルを格納するディレクトリ |

| ファイル又はディレクトリ名 | 説明 |
|---|---|
| %PDDIR%¥spool¥pdshmdump ※1 | 共用メモリダンプファイルを格納するディレクトリ |
| %PDDIR%¥spool¥pdlckinf ※1 | デッドロックタイムアウト情報ファイルを格納するディレクトリ |
| %PDDIR%¥spool¥pdsysdump ※1 | システム共通の簡易ダンプファイルを格納するディレクトリ |
| %PDDIR%¥spool¥pdsdsdump ※1 | シングルサーバ用の簡易ダンプファイルを格納するディレクトリ |
| %PDDIR%¥spool¥pdfesdump ※1 | フロントエンドサーバ用の簡易ダンプファイルを格納するディレクトリ |
| %PDDIR%¥spool¥pddicdump ※1 | ディクショナリサーバ用の簡易ダンプファイルを格納するディレクトリ |
| %PDDIR%¥spool¥pdbesdump ※1 | バックエンドサーバ用の簡易ダンプファイルを格納するディレクトリ |
| %PDDIR%¥spool¥pdstj1，pdstj2 | 統計ログファイル |
| %PDDIR%¥spool¥pdlog1，pdlog2 | メッセージログファイル |
| %PDDIR%¥spool¥pdjnlinf | システムログ情報出力ファイルを格納するディレクトリ |
| %PDDIR%¥spool¥pdjnlinf¥errinf | システムログエラー情報出力ファイルを格納するディレクトリ |
| %PDDIR%¥spool¥scdqid1，scdqid2 | HiRDB 内部のスケジュールキュー情報を格納するファイル |
| %PDDIR%¥spool¥oslmqid | メッセージキューの ID を格納するファイル |
| %PDDIR%¥spool¥oslsmid | セマフォの ID を格納するファイル |
| %PDDIR%¥spool¥pdprcsts | prc ステータスファイル |
| %PDDIR%¥spool¥~pdatmode | 開始・終了用ステータスファイル |
| %PDDIR%¥spool¥~pdipcid | セマフォの ID を管理するファイル |
| %PDDIR%¥spool¥~pdommenv | 共用メモリの情報を格納するファイル |
| %PDDIR%¥spool¥cmdlog¥cmdlog1，cmdlog2 | 実行したコマンドの履歴ファイル |
| %PDDIR%¥spool¥errlog¥errlog1，errlog2 | HiRDB の内部稼働履歴ファイル |
| %PDDIR%¥spool¥olkfifs | HiRDB 内部用ワークディレクトリ |
| %PDDIR%¥spool¥olkrsfs | HiRDB 内部用ワークディレクトリ |
| %PDDIR%¥spool¥cnctusrinf | 正常停止又は計画停止コマンド実行時に HiRDB に接続しているユーザがいた場合，接続ユーザ情報を格納するファイル |
| %PDDIR%¥spool¥cnctusrdtl | 正常停止又は計画停止コマンド実行時に HiRDB に接続しているユーザがいた場合，pdls -d act，pdls -d prc，pdls -d trn のコマンド実行結果を格納するファイル |
| %PDDIR%¥spool¥pdsqldump ※1 | アクセスパス情報ファイルを格納するディレクトリ |
| %PDDIR%¥spool¥pdrshs1.log，pdrshs2.log | リモート系コマンドの情報を格納するファイル |
| %PDDIR%¥spool¥pdsha1.log，pdsha2.log | 系切り替え機能の情報を格納するファイル |
| 任意※2 | RPC トレースファイル |
| %PDDIR%¥tmp ※3 | HiRDB 内部用ワークディレクトリ |
| %PDDIR%¥tmp¥pdommenv | 共用メモリの情報を格納するファイル |

| ファイル又はディレクトリ名 | 説明 |
|---|---|
| %PDDIR%¥tmp¥home¥HiRDB が管理する識別子のディレクトリ | カレントワーキングディレクトリ |
| %PDDIR%¥conf | HiRDB システム定義ファイルを格納するディレクトリ |
| %PDDIR%¥conf¥backconf | システム構成変更コマンド実行時の，変更前の HiRDB システム定義を格納するディレクトリ |
| %PDDIR%¥.dbenv | HiRDB データベース環境情報ファイルを格納するディレクトリ |
| %PDCLTPATH%¥pdsql1.trc，pdsql2.trc ※4 | UAP が実行した SQL のトレース情報を格納するファイル |
| %PDCLTPATH%¥pderr1.trc，pderr2.trc ※4 | UAP とサーバ間の通信エラー情報を格納するファイル |
| %PDDIR%¥plugin | HiRDB プラグイン運用ディレクトリを統括するディレクトリ |
| %PDDIR%¥plugin¥lib | プラグインライブラリを格納するディレクトリ |
| %PDDIR%¥plugin¥ プラグイン名称 | プラグイン運用ディレクトリ |
| %PDDIR%¥plugin¥ プラグイン名称 ¥bin | プラグイン専用コマンドを格納するディレクトリ |
| %PDDIR%¥plugin¥ プラグイン名称 ¥etc | 各プラグイン共通に必要なファイルを格納するディレクトリ |
| %PDDIR%¥plugin¥ プラグイン名称 ¥conf | プラグイン用コンフィグレーションファイルを格納するディレクトリ |
| %PDDIR%¥jre ※5 | Java™ 実行環境 |
| %PDDIR%¥renew | 修正版 HiRDB の入れ替え用ディレクトリ |
| %PDDIR%¥renew_bak | 修正版 HiRDB に入れ替えるときの，稼働中の HiRDB のバックアップ用ディレクトリ |
| %PDDIR%¥pdistup | 簡易セットアップツールを格納するディレクトリ |
| %PDDIR%¥sample | サンプルデータベースの実行環境を格納しているディレクトリ |
| %PDDIR%¥HiRDEF | HiRDB 定義支援の実行環境を格納しているディレクトリ |
| %PDDIR%¥UXPLDIR | システムコールの仕様差吸収ライブラリが使用するディレクトリ |
| %PDDIR%¥UXPLDIR¥spool¥uxpllog1.txt，uxpllog2.txt | システムコールの仕様差吸収ライブラリが出力する情報を格納するファイル |
| %PDDIR%¥Setup.ini | インストール情報を管理するファイル |

注※1

このディレクトリは，HiRDB がトラブルシュート情報を出力するディレクトリで，容量が増え続ける可能性があります。そのため，定期的に削除する必要があります。pdcspool コマンドで定期的に削除してください。

なお，次のオペランドでトラブルシュート情報を定期的に削除する設定ができます。オペランドの詳細は，マニュアル「HiRDB Version 8 システム定義」を参照してください。

- pd_spool_cleanup_interval
- pd_spool_cleanup_interval_level
- pd_spool_cleanup
- pd_spool_cleanup_level

注※2

pd_rpc_trace_name オペランドでファイル名を指定します。

注※3

HiRDB が内部的に使用するディレクトリです。このディレクトリ下にディレクトリ及びファイルを作成しないでください。また，HiRDB がファイルを作成するディレクトリにこのディレクトリを指定しないでください（例えば，pd_rpc_trace_name オペランドなど）。このディレクトリはユニットを開始するときに毎回削除及び新規作成されます。

注※4

このファイルは，PDCLTPATH で指定したディレクトリに二つ出力されます。PDCLTPATH の指定がない場合，UAP を起動したときのカレントディレクトリ（OpenTP1 から起動される UAP の場合，%DCDIR%¥tmp¥home¥ **サーバ名** xx のディレクトリ）下に出力されます。
作成されるファイル名は，X/Open に従った API（TX_ 関数）の使用の有無によって異なります。
TX_ 関数の使用時に作成されるファイル名は次のようになります。

- pdsqlxxxxx-1.trc，pdsqlxxxxx-2.trc
- pderrxxxxx-1.trc，pderrxxxxx-2.trc

（凡例）xxxxx：UAP 実行時のプロセス ID
ファイル名がプロセス ID になるため，UAP 実行時のサーバプロセス数分のファイルが出力される可能性があるので注意が必要です。

注※5

バージョン 07-03 より前の場合は作成されます。バージョン 07-03 以降の場合は，JRE が同梱されないため，作成されません。

## 1.2.2 単調増加ファイル

HiRDB を使用すると単調増加するファイルを情報の種別ごとに示します。

なお，* は任意の英数字です。また，ファイル名又はディレクトリ名には，標準のパス名を記載しています。システムによっては異なる場合があります。

また，サポートバージョンとは，サポートを開始したバージョンを示します。例えば，サポートバージョンが初期の場合は，全バージョンでサポートしていることになります。

表中の凡例を次に示します。

○：オプションなどで最大サイズを制限できます。
×：最大サイズを制限できません。
S：HiRDB/ シングルサーバ
P：HiRDB/ パラレルサーバ
DK：HiRDB/Developer's Kit
RT：HiRDB/Run Time

## (1) 簡易ダンプ

| 項番 | ファイル名又はディレクトリ名 | 説明 | ファイルサイズの概算値 | ファイル数 | 最大サイズの制限可否 | 出力される製品種別 | サポートバージョン |
|---|---|---|---|---|---|---|---|
| 1 | %PDDIR%¥spool¥pdfesdump¥* | フロントエンドサーバ用の簡易ダンプファイルです。pdfes プロセスセグメンテーション障害，又はアボート時に生成されます。<br>自動で削除する場合：<br>次のオペランドを指定します。<br>• pd_spool_cleanup_interval<br>• pd_spool_cleanup_interval_level<br>• pd_spool_cleanup<br>• pd_spool_cleanup_level<br>手動で削除する場合：<br>pdcspool コマンドを実行します。<br>なお，%PDDIR% 下が容量不足となった場合はユニットダウンします。 | 数 MB（不定） | ディレクトリ：<br>%PDDIR%¥spool¥pdfesdump[1-3] でディレクトリ 3 世代のループ<br>ファイル：<br>ディレクトリ下に無制限 | × | P | 初期 |
| 2 | %PDDIR%¥spool¥pddicdump¥* | ディクショナリサーバ用の簡易ダンプファイルです。pddic プロセスセグメンテーション障害，アボート，ディクショナリ用 RD エリア閉塞時に生成されます。<br>自動で削除する場合：<br>次のオペランドを指定します。<br>• pd_spool_cleanup_interval<br>• pd_spool_cleanup_interval_level<br>• pd_spool_cleanup<br>• pd_spool_cleanup_level<br>手動で削除する場合：<br>pdcspool コマンドを実行します。<br>なお，%PDDIR% 下が容量不足となった場合はユニットダウンします。 | 数 MB（不定） | ディレクトリ：<br>%PDDIR%¥spool¥pddicdump[1-3] でディレクトリ 3 世代のループ<br>ファイル：<br>ディレクトリ下に無制限 | × | P | 初期 |
| 3 | %PDDIR%¥spool¥pdbesdump¥* | バックエンドサーバ用の簡易ダンプファイルです。pdbes プロセスセグメンテーション障害，アボート，ユーザ用 RD エリア閉塞時に生成されます。<br>自動で削除する場合：<br>次のオペランドを指定します。<br>• pd_spool_cleanup_interval<br>• pd_spool_cleanup_interval_level<br>• pd_spool_cleanup<br>• pd_spool_cleanup_level<br>手動で削除する場合：<br>pdcspool コマンドを実行します。<br>なお，%PDDIR% 下が容量不足となった場合はユニットダウンします。 | 数 MB（不定） | ディレクトリ：<br>%PDDIR%¥spool¥pdbesdump[1-3] でディレクトリ 3 世代のループ<br>ファイル：<br>ディレクトリ下に無制限 | × | P | 初期 |

| 項番 | ファイル名又はディレクトリ名 | 説明 | ファイルサイズの概算値 | ファイル数 | 最大サイズの制限可否 | 出力される製品種別 | サポートバージョン |
|---|---|---|---|---|---|---|---|
| 4 | %PDDIR%¥spool¥pdsdsdump¥* | シングルサーバ用の簡易ダンプファイルです。pdsds プロセスセグメンテーション障害，アボート，ユーザ用 RD エリア閉塞時に生成されます。<br>自動で削除する場合：<br>次のオペランドを指定します。<br>• pd_spool_cleanup_interval<br>• pd_spool_cleanup_interval_level<br>• pd_spool_cleanup<br>• pd_spool_cleanup_level<br>手動で削除する場合：<br>pdcspool コマンドを実行します。<br>なお，%PDDIR% 下が容量不足となった場合はユニットダウンします。 | 数 MB（不定） | ディレクトリ：<br>%PDDIR%¥spool¥pdsdsdump[1-3] でディレクトリ 3 世代のループ<br>ファイル：<br>ディレクトリ下に無制限 | × | S | 初期 |
| 5 | %PDDIR%¥spool¥pdsysdump¥* | システム共通の簡易ダンプファイルです。HiRDB システムを制御するプロセスの異常終了時に生成されます。<br>自動で削除する場合：<br>次のオペランドを指定します。<br>• pd_spool_cleanup_interval<br>• pd_spool_cleanup_interval_level<br>• pd_spool_cleanup<br>• pd_spool_cleanup_level<br>手動で削除する場合：<br>pdcspool コマンドを実行します。<br>なお，%PDDIR% 下が容量不足となった場合はユニットダウンします。 | 数～数十 MB（不定） | ディレクトリ：<br>%PDDIR%¥spool¥pdsdsdump[1-3] でディレクトリ 3 世代のループ<br>ファイル：<br>ディレクトリ下に無制限 | × | S，P | 初期 |

## (2) エラー情報

| 項番 | ファイル名又はディレクトリ名 | 説明 | ファイルサイズの概算値 | ファイル数 | 最大サイズの制限可否 | 出力される製品種別 | サポートバージョン |
|---|---|---|---|---|---|---|---|
| 1 | %PDDIR%¥spool¥pdlckinf¥* | デッドロックタイムアウト情報ファイルです。排他制御エラー発生時に生成されます。<br>自動で削除する場合：<br>次のオペランドを指定します。<br>• pd_spool_cleanup_interval<br>• pd_spool_cleanup_interval_level<br>• pd_spool_cleanup<br>• pd_spool_cleanup_level<br>手動で削除する場合：<br>pdcspool コマンドを実行します。<br>なお，%PDDIR% 下が容量不足となった場合はユニットダウンします。 | 数 KB | 無制限 | × | S，P | 初期 |
| 2 | %PDDIR%¥spool¥pdjnlinf¥* | システムログファイルの空き容量監視機能のシステムログファイルの状態情報ファイルです。システムログファイルの空き容量が警告値未満になったときに 1 ファイル生成されます。<br>自動で削除する場合：<br>次のオペランドを指定します。<br>• pd_spool_cleanup_interval<br>• pd_spool_cleanup_interval_level<br>• pd_spool_cleanup<br>• pd_spool_cleanup_level<br>手動で削除する場合：<br>pdcspool コマンドを実行します。<br>なお，%PDDIR% 下が容量不足となった場合はユニットダウンします。 | 2729 ～ 3521 バイト程度 | サーバごとに 1 ファイル | × | S，P | 07-00 |

| 項番 | ファイル名又はディレクトリ名 | 説明 | ファイルサイズの概算値 | ファイル数 | 最大サイズの制限可否 | 出力される製品種別 | サポートバージョン |
|---|---|---|---|---|---|---|---|
| 3 | %PDDIR%¥spool¥pdjnlinf¥errinf¥* | システムログエラー情報出力ファイルです。リラン時，システムログ読み込みエラー時に生成されます。<br>自動で削除する場合：<br>次のオペランドを指定します。<br>• pd_spool_cleanup_interval<br>• pd_spool_cleanup_interval_level<br>• pd_spool_cleanup<br>• pd_spool_cleanup_level<br>手動で削除する場合：<br>pdcspool コマンドを実行します。又は pdstop コマンド実行時（オプションなし，-i オプション指定時，又は dbdestroy オプション指定時）にも削除されます。<br>なお，%PDDIR% 下が容量不足となった場合はユニットダウンします。 | 最大値は pd_log_max_data_size オペランドの指定値 | ログ世代数分（有限個） | × | S，P | 初期 |
| 4 | %PDCLTPATH%¥pderr1.trc，pderr2.trc | クライアントエラー情報ファイルです。UAP 実行時に生成されます。OS の del コマンドなどを実行して手動で削除します。 | クライアント環境変数 PDUAPERLOG で指定（省略値は 4096 バイト） | 2 ファイルをラップで使用 | ○ | S，P，DK，RT | 初期 |
| 5 | %PDCLTPATH%¥pderrxxxxx-1.trc，pderrxxxxx-2.trc（xxxxx：UAP 実行時のプロセス ID） | クライアントエラー情報ファイルです<br>（X/Open に従った API（TX_ 関数）を使用した場合）。UAP 実行時に生成されます。OS の del コマンドなどを実行して手動で削除します。 | クライアント環境変数 PDUAPERLOG で指定（省略値は 4096 バイト） | UAP のプロセス ID ごとに 2 ファイルを作成し，ラップで使用 | ○ | S，P，DK，RT | 07-01 |

## (3) チューニング情報

| 項番 | ファイル名又はディレクトリ名 | 説明 | ファイルサイズの概算値 | ファイル数 | 最大サイズの制限可否 | 出力される製品種別 | サポートバージョン |
|---|---|---|---|---|---|---|---|
| 1 | %PDDIR%¥spool¥pdsqldump¥* | アクセスパス情報ファイルです。<br>クライアント環境変数 PDVWOPTMODE に 1 以上を指定している場合，SQL 文実行時（前処理ごと）に生成されます。<br>自動で削除する場合：<br>次のオペランドを指定します。<br>• pd_spool_cleanup_interval<br>• pd_spool_cleanup_interval_level<br>• pd_spool_cleanup<br>• pd_spool_cleanup_level<br>手動で削除する場合：<br>pdcspool コマンドを実行します。<br>なお，%PDDIR% 下が容量不足となった場合はユニットダウンします。 | （数 KB ～数百 KB）× トランザクション内 SQL 数 | トランザクションごと | × | S，P | 04-03 |

## (4) トラブルシュート情報

| 項番 | ファイル名又はディレクトリ名 | 説明 | ファイルサイズの概算値 | ファイル数 | 最大サイズの制限可否 | 出力される製品種別 | サポートバージョン |
|---|---|---|---|---|---|---|---|
| 1 | %PDDIR%¥spool¥save¥abcode.* | サーバプロセス異常終了情報（アボートコードなど）です。サーバプロセスの異常終了時に生成されます。<br>自動で削除する場合：<br>次のオペランドを指定します。<br>• pd_spool_cleanup_interval<br>• pd_spool_cleanup_interval_level<br>• pd_spool_cleanup<br>• pd_spool_cleanup_level<br>手動で削除する場合：<br>pdcspool コマンドを実行します。<br>なお，%PDDIR% 下が容量不足となった場合はユニットダウンします。 | 32 ビットモードの場合：40 バイト<br>64 ビットモードの場合：72 バイト | 無制限 | × | S，P | 初期 |

| 項番 | ファイル名又はディレクトリ名 | 説明 | ファイルサイズの概算値 | ファイル数 | 最大サイズの制限可否 | 出力される製品種別 | サポートバージョン |
|---|---|---|---|---|---|---|---|
| 2 | %PDDIR%¥spool¥save¥* | サーバプロセス異常終了情報（コアファイル）です。サーバプロセスの異常終了時に生成されます。<br>自動で削除する場合：<br>次のオペランドを指定します。<br>• pd_spool_cleanup_interval<br>• pd_spool_cleanup_interval_level<br>• pd_spool_cleanup<br>• pd_spool_cleanup_level<br>手動で削除する場合：<br>pdcspool コマンドを実行します。<br>なお，%PDDIR% 下が容量不足となった場合はユニットダウンします。 | 数 MB ～数十 MB | ダウンしたサーバ名 [1-3] でファイル 3 世代のループ | × | S，P | 初期 |
| 3 | %PDDIR%¥spool¥save¥*.deb | サーバプロセス異常終了情報です。サーバプロセスの異常終了時に生成されます。<br>自動で削除する場合：<br>次のオペランドを指定します。<br>• pd_spool_cleanup_interval<br>• pd_spool_cleanup_interval_level<br>• pd_spool_cleanup<br>• pd_spool_cleanup_level<br>手動で削除する場合：<br>pdcspool コマンドを実行します。<br>なお，%PDDIR% 下が容量不足となった場合はユニットダウンします。 | 数 MB ～十数 MB | ダウンしたサーバ名 [1-3].deb でファイル 3 世代のループ | × | S，P | 初期 |
| 4 | %PDDIR%¥spool¥save¥ *コマンド名*.txt | pdload，pdrorg，pdrreclaim，pdpgbfon コマンドのトラブルシュート情報です。コマンドのタイムアウト時に生成されます。<br>自動で削除する場合：<br>次のオペランドを指定します。<br>• pd_spool_cleanup_interval<br>• pd_spool_cleanup_interval_level<br>• pd_spool_cleanup<br>• pd_spool_cleanup_level<br>手動で削除する場合：<br>pdcspool コマンドを実行します。<br>なお，%PDDIR% 下が容量不足となった場合はユニットダウンします。 | 数 KB ～数十 KB（そのときに実行していた HiRDB のプロセス数や，排他待ち資源数によって変化します） | コマンドのプロセス ID ごとに 1 ファイル | × | S，P | 06-02 |

| 項番 | ファイル名又はディレクトリ名 | 説明 | ファイルサイズの概算値 | ファイル数 | 最大サイズの制限可否 | 出力される製品種別 | サポートバージョン |
|---|---|---|---|---|---|---|---|
| 5 | %PDDIR%¥spool¥pdshmdump¥* | 共用メモリダンプファイルです。<br>サーバプロセス異常終了時，ユニット異常終了時に生成されます。<br>自動で削除する場合：<br>次のオペランドを指定します。<br>• pd_spool_cleanup_interval<br>• pd_spool_cleanup_interval_level<br>• pd_spool_cleanup<br>• pd_spool_cleanup_level<br>手動で削除する場合：<br>pdcspool コマンドを実行します。<br>なお，%PDDIR% 下が容量不足となった場合はユニットダウンします。 | 数百 MB ～数 GB（pdls -d mem コマンドで共用メモリを使用するプロセスの属性が "MANAGER" と表示された共用メモリセグメントのサイズとほぼ同じです） | shmdump と shmdump.old のファイル 2 世代でループ | × | S，P | 初期 |

## (5) インデクス作成用一時ファイル

| 項番 | ファイル名又はディレクトリ名 | 説明 | ファイルサイズの概算値 | ファイル数 | 最大サイズの制限可否 | 出力される製品種別 | サポートバージョン |
|---|---|---|---|---|---|---|---|
| 1 | *pd_plugin_ixmk_dir 指定ディレクトリ¥インデクス名称 .RD エリア名称*（pd_plugin_ixmk_dir にディレクトリを指定した場合） | プラグインインデクスのインデクス情報ファイルです。SQL 文実行時に生成されます。<br>プラグインインデクスの一括作成が正常終了したとき，プラグインインデクスの再作成を実行したときに，自動で削除されます。<br>手動で削除する場合は OS の del コマンドなどを実行します。 | マニュアル「HiRDB Version 8 システム運用ガイド」の「プラグインインデクスの遅延一括作成」の「注意事項」を参照してください。 | プラグイン数×インデクス格納 RD エリア数（更新の発生した RD エリア数） | × | S，P | 05-06 |

## (6) 運用コマンド作業用一時ファイル

| 項番 | ファイル名又はディレクトリ名 | 説明 | ファイルサイズの概算値 | ファイル数 | 最大サイズの制限可否 | 出力される製品種別 | サポートバージョン |
|---|---|---|---|---|---|---|---|
| 1 | %PDDIR%¥tmp¥CMr* | 運用コマンド一時ファイルです。<br>運用コマンド実行時に生成されます。<br>運用コマンド実行終了時に自動的に削除されます。 | MAX（1024 バイト×グローバルバッファ数，512 バイト×RD エリア数） | コマンドのプロセス ID ごとに 1 ファイル | ○ | S，P | 初期 |
| 2 | %PDDIR%¥tmp¥CMs* | 運用コマンド一時ファイルです。<br>運用コマンド実行時に生成されます。<br>運用コマンド実行終了時に自動的に削除されます。 | 128 バイト×RD エリア数 | コマンドのプロセス ID ごとに 1 ファイル | ○ | S，P | 初期 |
| 3 | %PDDIR%¥tmp¥CMb* | pdbufls コマンドの差分情報ファイルです。pdbufls コマンド（-k sts 指定，又はオプション指定なし）実行時に生成されます。<br>HiRDB 開始時に自動で削除されます。手動で削除する場合は，pdbufls コマンドを実行していないときに OS の del コマンドなどを実行します。 | 16 バイト＋124 バイト×グローバルバッファ数 | コマンドのプロセス ID ごとに 1 ファイル | ○ | S，P | 初期 |
| 4 | %PDDIR%¥tmp¥pdcmd* | 運用コマンド結果ファイルです。<br>運用コマンド実行時に生成されます。運用コマンド実行終了時に自動で削除されます。<br>なお，このファイルは HiRDB/ パラレルサーバでディクショナリサーバとシステムマネジャが別ノードの場合にだけ使用されます。 | MAX（1024 バイト×グローバルバッファ数，512 バイト×RD エリア数） | コマンドのプロセス ID ごとに 1 ファイル | ○ | P | 初期 |
| 5 | %TMP%¥pddefrev.exp.*（pd_tmp_directory オペランドは無効です） | pddefrev コマンド作業用一時ファイルです。pddefrev コマンド実行開始時に生成されます。<br>pddefrev コマンド実行終了時に自動で削除されます。手動で削除する場合は，OS の del コマンドなどを実行します。 | 搬出対象の資源数×70 バイト | コマンドのプロセス ID ごとに 1 ファイル | × | S，P | 04-01 |
| 6 | %PDDIR%¥spool¥tmp¥* | 運用コマンド一時ファイルです。<br>運用コマンド実行時に生成されます。<br>運用コマンド実行終了時に自動的に削除されます。 | 最大 1024 バイト | コマンドのプロセス ID ごとに 1 ファイル | × | S，P | 08-51-16 または 08-51-/P |

## (7) ユティリティ結果ファイル

| 項番 | ファイル名又はディレクトリ名 | 説明 | ファイルサイズの概算値 | ファイル数 | 最大サイズの制限可否 | 出力される製品種別 | サポートバージョン |
|---|---|---|---|---|---|---|---|
| 1 | %TMP%¥pdcp1* | pdcopy 結果ファイルです。pdcopy 実行時に生成されます。<br>OS の del コマンドなどを実行して手動で削除します。 | 700KB × RD エリア数 | pdcopy の -p オプションなしの場合に毎回異なるファイル名で 1 ファイル | ○ | S，P | 初期 |
| 2 | %TMP%¥pdrs1* | pdrstr 結果一時ファイルです（ログ指定回復時のロールバックログ格納一時ファイル）。pdrstr 実行時に生成されます。<br>ロールバック完了時に自動で削除されます。手動で削除する場合は，OS の del コマンドなどを実行します。 | ロールバックログ量に依存します。 | pdrstr の対象 RD エリアが属するサーバ数 | × | S，P | 初期 |
| 3 | %TMP%¥pdrs2* | pdrstr 結果ファイルです。pdrstr 実行時に生成されます。<br>OS の del コマンドなどを実行して手動で削除します。 | 700KB × RD エリア数 | pdrstr の -p オプションなしの場合に毎回異なるファイル名で 1 ファイル | ○ | S，P | 初期 |
| 4 | %TMP%¥pdrs4* | pdrstr 結果一時ファイルです（メッセージをコンソールに出力するための一時ファイル）。pdrstr 実行時に生成されます。<br>pdrstr 実行終了時に自動で削除されます。手動で削除する場合は，OS の del コマンドなどを実行します。 | 256 バイト | 1（コマンド実行ノードだけ） | ○ | S，P | 初期 |
| 5 | %TMP%¥REPORT*（バージョン 08-02 以降で pd_tmp_directory オペランド指定時は pd_tmp_directory ディレクトリ下） | pdrbal 処理結果ファイルです。pdrbal 実行時に生成されます。<br>OS の del コマンドなどを実行して手動で削除します。 | 「19.1.9 リバランスユティリティ（pdrbal）実行時のファイルの容量」を参照してください。 | report 制御文指定なしの場合に毎回異なるファイル名で 1 ファイル | × | S，P | 06-00 |

| 項番 | ファイル名又はディレクトリ名 | 説明 | ファイルサイズの概算値 | ファイル数 | 最大サイズの制限可否 | 出力される製品種別 | サポートバージョン |
|---|---|---|---|---|---|---|---|
| 6 | %TMP%¥CONSTCK-REPORT-*（08-02 以降で pd_tmp_directory オペランド指定時は pd_tmp_directory ディレクトリ下） | pdconstck 処理結果ファイルです。pdconstck 実行開始時に生成されます。<br>OS の del コマンドなどを実行して手動で削除します。 | 「19.1.10　整合性チェックユティリティ（pdconstck）実行時のファイルの容量」を参照してください。 | コマンドのプロセス ID ごとに 1 ファイル | × | S，P | 07-03 |

## (8) ユティリティ作業用一時ファイル

| 項番 | ファイル名又はディレクトリ名 | 説明 | ファイルサイズの概算値 | ファイル数 | 最大サイズの制限可否 | 出力される製品種別 | サポートバージョン |
|---|---|---|---|---|---|---|---|
| 1 | %TMP%¥pdcp3* | pdcopy 結果一時ファイル（メッセージをコンソールに出力するための一時ファイル）です。pdcopy 実行時に生成されます。<br>pdcopy 実行終了時に自動的に削除されます。手動で削除する場合は，OS の del コマンドなどを実行します。 | 256 バイト | 1（コマンド実行ノードだけ） | ○ | S，P | 初期 |
| 2 | %TMP%¥pdcp4* | pdcopy 結果一時ファイル（メッセージ格納一時ファイル）です。pdcopy 実行開始時に生成されます。<br>pdcopy 実行終了時に自動的に削除されます。手動で削除する場合は，OS の del コマンドなどを実行します。 | 256 バイト×pdcopy 対象 RD エリア数 | pdcopy の対象 RD エリアが属するサーバ数＋ 2 | ○ | S，P | 初期 |
| 3 | %TMP%¥pdrs5* | pdrstr 結果一時ファイル（メッセージ格納一時ファイル）です。pdrsrt 実行開始時に生成されます。<br>pdrstr 実行終了時に自動的に削除されます。手動で削除する場合は，OS の del コマンドなどを実行します。 | 256 バイト×pdcopy 対象 RD エリア数 | 対象 RD エリアが属するサーバ数＋ 2<br>ただし，対象サーバ数が 2 以上，又は回復対象ユニット以外に存在するログを入力する場合には，さらに +1。 | ○ | S，P | 初期 |

| 項番 | ファイル名又はディレクトリ名 | 説明 | ファイルサイズの概算値 | ファイル数 | 最大サイズの制限可否 | 出力される製品種別 | サポートバージョン |
|---|---|---|---|---|---|---|---|
| 4 | %TMP%¥ERROR*（08-02 以降で pd_tmp_directory オペランド指定時は pd_tmp_directory ディレクトリ下） | pdload 入力データ格納情報の結果ファイル（エラー情報ファイル）です。pdload 実行時に生成されます。OS の del コマンドなどを実行して手動で削除します。 | 「19.1.1 データベース作成ユティリティ（pdload）実行時のファイルの容量」を参照してください。 | source 文で error オペランドを指定しない場合，毎回異なるファイル名で 1 ファイル | × | S，P | 初期 |
| 5 | %TMP%¥*（08-02 以降で pd_tmp_directory オペランド指定時は pd_tmp_directory ディレクトリ下） | pdload エラー情報ファイル作成用一時ファイルです。pdload 実行時に生成されます。OS の del コマンドなどを実行して手動で削除します。<br>なお，このファイルは HiRDB/パラレルサーバの場合だけ使用されます。 | 「19.1.1 データベース作成ユティリティ（pdload）実行時のファイルの容量」を参照してください。 | データロード対象表格納サーバ数 | × | S，P | 初期 |
| 6 | %TMP%¥LOBMID*（08-02 以降で pd_tmp_directory オペランド指定時は pd_tmp_directory ディレクトリ下） | pdload BLOB 列ロード用ワークファイルです。pdload 実行開始時に生成されます。OS の del コマンドなどを実行して手動で削除します。 | 「19.1.1 データベース作成ユティリティ（pdload）実行時のファイルの容量」を参照してください。 | オプション -k に d 以外を指定し，lobmid 文を省略して BLOB 列を持つ表にデータロードする場合に，毎回異なるファイル名で 1 ファイル | × | S，P | 03-00 |

| 項番 | ファイル名又はディレクトリ名 | 説明 | ファイルサイズの概算値 | ファイル数 | 最大サイズの制限可否 | 出力される製品種別 | サポートバージョン |
|---|---|---|---|---|---|---|---|
| 7 | %TMP%¥INDEX*,<br>*idxwork制御文指定ディレクトリ*¥INDEX*（08-02以降でpd_tmp_directoryオペランド指定時はpd_tmp_directoryディレクトリ下） | pdload，pdrorg，pdrbal のインデクス情報ファイルです。pdload，pdrorg，pdrbal 実行時に生成されます。<br>インデクスロード完了時に自動的に削除されます。手動で削除する場合は，OS の del コマンドなどを実行します。 | 次の箇所を参照してください。<br>「19.1.1 データベース作成ユティリティ（pdload）実行時のファイルの容量」<br>「19.1.2 データベース再編成ユティリティ（pdrorg）実行時のファイルの容量」<br>「19.1.9 リバランスユティリティ（pdrbal）実行時のファイルの容量」 | インデクス一括作成モード選択時，毎回異なるファイル名でインデクス数×インデクス格納 RD エリア数分 | × | S，P | 初期 |

| 項番 | ファイル名又はディレクトリ名 | 説明 | ファイルサイズの概算値 | ファイル数 | 最大サイズの制限可否 | 出力される製品種別 | サポートバージョン |
|---|---|---|---|---|---|---|---|
| 8 | %TMP%¥rs*,<br>*sort 制御文指定ディレクトリ*¥rs*<br>（08-02 以降で pd_tmp_directory オペランド指定時は pd_tmp_directory ディレクトリ下） | pdload，pdrorg，pdrbal のインデクス情報ファイルです。pdload，pdrorg，pdrbal 実行時に生成されます。<br>インデクスロード完了時に自動的に削除されます。手動で削除する場合は，OS の del コマンドなどを実行します。 | 次の箇所を参照してください。<br>「19.1.1　データベース作成ユティリティ（pdload）実行時のファイルの容量」<br>「19.1.2　データベース再編成ユティリティ（pdrorg）実行時のファイルの容量」<br>「19.1.9　リバランスユティリティ（pdrbal）実行時のファイルの容量」 | インデクス一括作成モード選択時，毎回異なるファイル名でインデクス格納サーバ数分 | × | S，P | 初期 |
| 9 | %TMP%¥*.dbst.data,<br>%TMP%¥*.dbst.msg<br>（08-02 以降で pd_tmp_directory オペランド指定時は pd_tmp_directory ディレクトリ下） | pddbst コマンド一時ファイルです。pddbst 実行開始時に生成されます。<br>pdrstr 実行終了時に自動的に削除されます。手動で削除する場合は，OS の del コマンドなどを実行します。 | 「19.1.4　データベース状態解析ユティリティ（pddbst）実行時のファイルの容量」を参照してください。 | コマンドのプロセス ID ごとにそれぞれ 1 ファイル | × | S，P | 初期 |

| 項番 | ファイル名又はディレクトリ名 | 説明 | ファイルサイズの概算値 | ファイル数 | 最大サイズの制限可否 | 出力される製品種別 | サポートバージョン |
|---|---|---|---|---|---|---|---|
| 10 | %TMP%¥*.syi,<br>*.syo,<br>*.uai,<br>*.uao,<br>*.sqi,<br>*.sqo,<br>*.pci,<br>*.pco,<br>*.soi,<br>*.soo,<br>*.doi,<br>*.doo,<br>*.bui,<br>*.buo,<br>*.fii,<br>*.fio,<br>*.dfi,<br>*.dfo,<br>*.ixi,<br>*.ixo,<br>*.isi,<br>*.iso,<br>*.cni,<br>*.cno,<br>*.qhi,<br>*.qho,<br>*.shi,<br>*.sho,<br>*.obi,<br>*.obo,<br>*.fsi,<br>*.fso,<br>*.hbi,<br>*.hbo，*-w* オプション指定ディレクトリ¥***<br>（08-02 以降で pd_tmp_directory オペランド指定時は pd_tmp_directory ディレクトリ下） | pdstedit コマンド作業用一時ファイルです。pdstedit 実行開始時に生成されます。<br>pdstedit 実行終了時に自動的に削除されます。手動で削除する場合は，OS の del コマンドなどを実行します。 | 「19.1.3 統計解析ユティリティ（pdstedit）実行時のファイルの容量」を参照してください。 | コマンドのプロセス ID ごと，かつ解析対象の情報ごとに 1 ファイル | × | S，P | 初期 |

## (9) トレース情報

| 項番 | ファイル名又はディレクトリ名 | 説明 | ファイルサイズの概算値 | ファイル数 | 最大サイズの制限可否 | 出力される製品種別 | サポートバージョン |
|---|---|---|---|---|---|---|---|
| 1 | %PDCLTPATH%¥pdsql1.trc,pdsql2.trc | SQL トレース情報です。SQL 文実行時に生成されます。<br>OS の del コマンドなどを実行して手動で削除します。 | クライアント環境変数 PDSQLTRACE で指定 | 2 ファイルをラップで使用 | ○ | S，P，DK，RT | 初期 |
| 2 | %PDCLTPATH%¥pdsqlxxxxx-1.trc,pdsqlxxxxx-2.trc<br>xxxxx：UAP 実行時のプロセス ID） | SQL トレース情報（X/Open に従った API（TX_ 関数）を使用した場合）です。SQL 文実行時に生成されます。<br>OS の del コマンドなどを実行して手動で削除します。 | クライアント環境変数 PDSQLTRACE で指定 | UAP のプロセス ID ごとに 2 ファイルを作成し，ラップで使用 | ○ | S，P，DK，RT | 07-01 |
| 3 | %PDCLTPATH%¥pdjsqlxxxxxxx_ppppp_1.trc,pdjsqlxxxxxxx_ppppp_2.trc<br>xxxxxxxx：接続したサーバ名<br>ppppp：クライアント側の受信ポート番号 | SQL トレース情報（Type4 JDBC ドライバを使用した場合）です。SQL 文実行時に生成されます。<br>OS の del コマンドなどを実行して手動で削除します。 | クライアント環境変数 PDSQLTRACE で指定 | 接続したサーバ名とクライアント側の受信ポート番号ごとに 2 ファイルを作成し，ラップで使用 | ○ | S，P，DK，RT | 08-00 |
| 4 | %PDCLTPATH%¥pdjsqlxxxxxxx_ppppp_1.trc,pdjsqlxxxxxxx_ppppp_2.trc<br>xxxxxxxx：接続したサーバ名<br>ppppp：クライアント側の受信ポート番号 | 動的 SQL トレース（Type4 JDBC ドライバを使用した場合）です。SQL 文実行時に生成されます。<br>OS の del コマンドなどを実行して手動で削除します。 | pdtrcmgr -s コマンドで指定 | 接続したサーバ名とクライアント側の受信ポート番号ごとに 2 ファイルを作成し，ラップで使用 | ○ | S，P，DK，RT | 08-00 |

| 項番 | ファイル名又はディレクトリ名 | 説明 | ファイルサイズの概算値 | ファイル数 | 最大サイズの制限可否 | 出力される製品種別 | サポートバージョン |
|---|---|---|---|---|---|---|---|
| 5 | pdoletrc.txt | OLEDB メソッドトレースです。<br>OLEDB プロバイダアクセス時に生成されます。<br>OS の del コマンドなどを実行して手動で削除します。 | サイズ指定はありません | 1 ファイル | × | S，P，DK，RT | 05-06 |
| 6 | %PDJDBFILEDIR%¥pdexc1.trc，pdexc2.trc | Exception トレースログです。<br>Type4 JDBC ドライバ内で例外発生時に生成されます。<br>OS の del コマンドなどを実行して手動で削除します。 | ↑ 180 × n × m ÷ 1024 ↑ ＋ 1 キロバイト<br>n：クライアント環境変数 PDJDBONMEMNUM で指定（省略値は 1000）<br>m：クライアント環境変数 PDJDBFILEOUTNUM で指定（省略値は 5） | 2 ファイルをラップで使用 | ○ | S，P，DK，RT | 08-00 |
| 7 | ユーザ指定（Driver Manager クラスの setLogWriter メソッド，DataSource インタフェースの setLogWriter メソッドで指定） | JDBC インタフェースメソッドトレースです。Type4 JDBC ドライバ内で例外発生時，Connection.close メソッド実行時に生成されます。<br>OS の del コマンドなどを実行して手動で削除します。 | ↑ 180 × n × m ÷ 1024 ↑ キロバイト<br>n：接続時に指定するユーザプロパティの TRC_NO で指定（省略値は 500）<br>m：接続から切断までの例外発生回数＋ 1 | 1 ファイル | ○ | S，P，DK，RT | 08-00 |

| 項番 | ファイル名又はディレクトリ名 | 説明 | ファイルサイズの概算値 | ファイル数 | 最大サイズの制限可否 | 出力される製品種別 | サポートバージョン |
|---|---|---|---|---|---|---|---|
| 8 | %PDTRCPATH%¥pdcHHMMSSmmm_XXX_1.trc, pdcHHMMSSmmm_XXX_2.trc HHMMSSmmm：CONNECT 時間 XXX：CONNECT 通番 | 動的 SQL トレースです。SQL 文実行時に生成されます。<br>OS の del コマンドなどを実行して手動で削除します。 | pdtrcmgr -s コマンドで指定 | コネクトごとに 2 ファイルを作成し，ラップで使用 | ○ | S，P，DK，RT | 06-00 |
| 9 | %PDCLTPATH%¥pdrcnct1.trc, pdrcnct2.trc | 再接続トレースです。自動再接続機能で自動的に接続したときに生成されます。<br>OS の del コマンドなどを実行して手動で削除します。 | クライアント環境変数 PDRCTRACE で指定 | 2 ファイルをラップで使用 | ○ | S，P，DK，RT | 07-01 |
| 10 | %PDCLTPATH%¥pddndpxxxx_yyy_1.trc, pddndpxxxx_yyy_2.trc xxxx：プロセス ID yyy：コネクト通番 | HiRDB.NET データプロバイダのメソッドトレースです。サーバとの接続中，HiRDB.NET データプロバイダのメソッド及びプロパティ呼び出し時に生成されます。<br>OS の del コマンドなどを実行して手動で削除します。 | クライアント環境変数 PDDNDPTRACE で指定 | 接続ごとに 2 ファイルをラップで使用 | ○ | DK，RT | 08-05 |

## (10) 統計情報

| 項番 | ファイル名又はディレクトリ名 | 説明 | ファイルサイズの概算値 | ファイル数 | 最大サイズの制限可否 | 出力される製品種別 | サポートバージョン |
|---|---|---|---|---|---|---|---|
| 1 | %PDREPPATH%¥pdHHMMSSmmm_XXX_1.trc,<br>pdHHMMSSmmm_XXX_2.trc<br>HHMMSSmmm：CONNECT 時間<br>XXX：CONNECT 通番 | UAP 統計レポートです。SQL 文実行時に生成されます。<br>OS の del コマンドなどを実行して手動で削除します。 | クライアント環境変数 PDSQLTRACE で指定 | コネクトごとに 2 ファイルを作成し，ラップで使用 | ○ | S，P，DK，RT | 06-00 |
| 2 | %PDCLTPATH%¥pdjsqlxxxxxxx_ppppp_1.trc,<br>pdjsqlxxxxxxx_ppppp_2.trc<br>xxxxxxxx：接続したサーバ名<br>ppppp：クライアント側の受信ポート番号 | UAP 統計レポート（Type4 JDBC ドライバを使用した場合）です。SQL 文実行時に生成されます。<br>OS の del コマンドなどを実行して手動で削除します。 | クライアント環境変数 PDSQLTRACE で指定 | 接続したサーバ名とクライアント側の受信ポート番号ごとに 2 ファイルを作成し，ラップで使用 | ○ | S，P，DK，RT | 08-00 |

# 1.3 HiRDB のバージョンアップ

HiRDB をバージョンアップする方法について説明します。

HiRDB/ パラレルサーバをバージョンアップする場合は，すべてのユニットをバージョンアップして，HiRDB/ パラレルサーバを構成するすべてのユニット間のバージョンを合わせてください。

**注意事項**

- HiRDB をバージョンアップするときに，**旧バージョンの** HiRDB **をアンインストールしないでください。**新バージョンの HiRDB を上書きインストールしてください。
- セキュリティ監査機能を使用している場合にバージョンアップするときは，注意事項があります。注意事項については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。
- Java ストアドプロシジャ及び Java ストアドファンクションを使用している場合に 07-03 以降にバージョンアップするときは，注意事項があります。注意事項については，「1.3.4　Java ストアドプロシジャ及び Java ストアドファンクションを使用する場合」を参照してください。

## 1.3.1 バージョンアップ前にすること

バージョンアップをする前に，次に示す内容を必ず実施してください。なお，マルチ HiRDB の場合，すべての運用ディレクトリに対して，次の内容を実施してください。

- HiRDB がオンライン状態であるかどうかの確認
- HiRDB の状態確認

以降，Windows の［コマンドプロンプト］の画面を使って，コマンドとユティリティを実行してください。

### (1) 空き領域の確認

データベース状態解析ユティリティ（pddbst）で，データディクショナリ用 RD エリアに必要な空き領域があるかどうかを確認してください。空き領域がない場合は，次に示すどちらかの方法で空き領域を確保してください。

- データベース再編成ユティリティ（pdrorg）でディクショナリ表を再編成します。
- データベース構成変更ユティリティ（pdmod）でデータディクショナリ用 RD エリアを拡張します。

この空き領域の確認は，旧バージョンと新バージョンを入れ替えるときにだけ必要です。修正版 HiRDB への入れ替えの場合は不要です。

ユティリティを実行する方法については，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

**バージョンアップに必要な空き領域**

バージョンアップをする前の HiRDB のバージョンに応じて，次の表に示す空き領域があるかどうかを確認してください。空き領域が不足していると，バージョンアップ後の HiRDB 開始時又は pdvrup コマンド実行時に容量不足でエラーとなることがあります。

表 1-5 バージョンアップするのに必要な空き領域

| データディクショナリ用 RD エリアに格納しているディクショナリ表 | データディクショナリ用 RD エリアに必要な空きセグメント数 | | | | |
|---|---|---|---|---|---|
| | 07-00 以降からバージョンアップする場合 | 06-00 以降からバージョンアップする場合 | 05-02 以降からバージョンアップする場合 | 03-00 以降からバージョンアップする場合 | 02-05 以前からバージョンアップする場合 |
| SQL_TABLES 表 | 1 | 3 | 3 | 3 | 3 |
| SQL_COLUMNS 表 | 4＋↑5÷S↑ | 5＋↑5÷S↑ | 5＋↑15÷S↑ | 5＋↑20÷S↑ | 5＋↑25÷S↑ |
| SQL_INDEXES 表 | 1 | 1 | 4 | 4 | 4 |
| SQL_TABLE_PRIVILEGES 表 | 1 | 1 | 3 | 3 | 3 |
| SQL_INDEX_COLINF 表 | 1 | 1 | 3 | 3 | 3 |
| SQL_VIEW_TABLE_USAGE 表 | 2 | 3 | 4 | 4 | 4 |
| SQL_VIEWS 表 | 2＋↑5÷S↑ | 3＋↑5÷S↑ | 3＋↑15÷S↑ | 3＋↑20÷S↑ | 3＋↑25÷S↑ |
| SQL_VIEW_DEF 表 | 2＋↑10÷S↑ | 3＋↑10÷S↑ | 2＋↑40÷S↑ | 2＋↑65÷S↑ | 2＋↑70÷S↑ |
| SQL_DIV_COLUMN 表※1 | － | － | － | 1 | － |
| SQL_ROUTINES 表※2 | － | － | 4＋↑30÷S↑ | 4＋↑30÷S↑ | － |
| SQL_ROUTINE_PARAMS 表※2 | － | － | 2＋↑20÷S↑ | 2＋↑20÷S↑ | － |

（凡例）－：該当しません。

S：対象表を格納するデータディクショナリ用 RD エリアのセグメントサイズ

注※1

LOB 列の定義がない場合又はデータディクショナリ LOB 用 RD エリアの定義がない場合は不要です。

注※2

ストアドプロシジャを定義していない場合は不要です。

## (2) システム用 RD エリアのバックアップの取得

データベース複写ユティリティ（pdcopy）で，すべての RD エリアのバックアップを取得してください。

**バックアップの取得手順**

1. pdstop コマンドで HiRDB を正常終了させます。
2. pdstart -r コマンドで HiRDB を開始します。
3. データベース複写ユティリティ（pdcopy）で RD エリアのバックアップを取得します。このとき，参照・更新不可能モード（-M x 指定）を指定してください。バックアップの取得方法については，マニュアル「HiRDB Version 8 システム運用ガイド」又は「HiRDB Version 8 コマンドリファレンス」を参照してください。

### (3) メモリ容量の確認

バージョンアップのコマンド（pdvrup コマンド）を実行するのに 135 キロバイトのメモリが必要です。メモリの空き容量を確認してください。なお，修正版 HiRDB への入れ替えの場合は不要です。

### (4) HiRDB がオンライン状態であるかどうかの確認

pdls コマンドですべてのユニットが ACTIVE と表示されているかどうかを確認してください。ACTIVE と表示されている場合，pdstop コマンドで正常終了させてください。

### (5) HiRDB の正常終了

バージョンアップする前に HiRDB を正常終了させてください。HiRDB/ パラレルサーバの場合はシステムマネジャがあるマシンから終了させてください。HiRDB が既に終了している場合は，次に示す情報を参照して HiRDB が正常終了しているかどうかを確認してください。

- **メッセージログファイル又はイベントログ**

正常終了していない場合は，pdstart コマンドでいったん HiRDB を開始して，pdstop コマンドで正常終了させてください。

### (6) HiRDB の状態確認

HiRDB をバージョンアップするユニットの状態を確認するため，pdls -d ust コマンドを実行します。

終了ステータスが 4 の場合（ユニットの状態が STARTING 又は STOPPING）：
HiRDB が開始処理の途中，又は停止処理の途中です。処理が終了してから pdls -d ust コマンドを再度実行してください。

終了ステータスが 8 の場合（ユニットの状態が PAUSE）：
障害によって，プロセスサービスの再起動を中断した状態です。KFPS00715-E メッセージ及びこのメッセージ以前のイベントログに出力されたメッセージを参照して障害の原因を取り除いてから，サービスを開始してください。その後，ユニットを再開始して，pdstop コマンドで正常終了させてください。

### (7) HiRDB のサービスの停止

バージョンアップする前にバージョンアップする HiRDB のサービスを停止してください。サービスの停止方法及び確認方法については，OS のマニュアルを参照してください。

### (8) コマンド，ユティリティ，アプリケーション，HiRDB Datareplicator，及び HiRDB Dataextractor の停止

コマンド，ユティリティ，アプリケーション，HiRDB Datareplicator，及び HiRDB Dataextractor はあらかじめ停止しておいてください。これらを停止しないと，実行形式ファイルや共用ライブラリの削除に失敗し，バージョンアップに失敗することがあります。

### (9) メモリ所要量の確認

HiRDB をバージョンアップすると，HiRDB のメモリ所要量が増えることがあります。「15. HiRDB のメモリ所要量」を参照して HiRDB のメモリ所要量を確認してください。

### (10) ステータスファイルの容量の確認

HiRDB をバージョンアップすると，HiRDB のステータスファイル容量が増えることがあります。「17.3

ステータスファイルの容量の見積もり」を参照して HiRDB のステータスファイルの容量を確認してください。

### (11) シンクポイントダンプファイルの容量の確認

HiRDB をバージョンアップすると，HiRDB のシンクポイントダンプファイルの容量が増えることがあります。「17.2　シンクポイントダンプファイルの容量の見積もり」を参照して HiRDB のシンクポイントダンプファイルの容量を確認してください。

### (12) システムログファイルの総レコード数の確認

バージョンアップする場合は，上書きできる状態のシステムログファイルの総レコード数を確認してください。次に示す条件式を満たさないと，バージョンアップに失敗することがあります。

**HiRDB Version 4.0 以前からバージョンアップする場合**

総レコード数※ ＞ 5754880 ÷システムログファイルのレコード長

**HiRDB Version 5.0 以前からバージョンアップする場合**

総レコード数※ ＞ 4239360 ÷システムログファイルのレコード長

**HiRDB Version 6 以前からバージョンアップする場合**

総レコード数※ ＞ 3215360 ÷システムログファイルのレコード長

**HiRDB Version 7 以前からバージョンアップする場合**

総レコード数※ ＞ 1413120 ÷システムログファイルのレコード長

なお，HiRDB/ パラレルサーバの場合は，ディクショナリサーバのシステムログファイル（上書きできる状態）の総レコード数を確認してください。

注※

次に示すどちらかの方法でシステムログファイルの総レコード数を確認してください。

- pdloginit コマンドの -n オプションの指定値の合計が総レコード数となります。
- pdlogls -d sys -s **サーバ名** -e コマンドを実行してください。実行結果の Recode-count の先頭部分に出力されたレコード数（16 進数）の合計が総レコード数となります。

### (13) 付加 PP のバージョンアップ

バージョンアップ前の HiRDB で付加 PP を使用していた場合，HiRDB と同じバージョンの付加 PP をインストールする必要があります。付加 PP については，「2.4　付加 PP のインストール時の注意」を参照してください。

### (14) HiRDB システム定義のオペランドの省略値の確認

バージョン 07-00 と 08-00 で，HiRDB システム定義の一部のオペランドの省略値を変更（省略値を推奨値に変更）しました。また，幾つかのオペランドについては省略できるようにしました。

そのため，例えば，07-03 から 08-02 にバージョンアップする場合，省略値が変更されているオペランドがあるため，HiRDB の稼働環境がバージョンアップによって変わることがあります。省略値の変更によって問題が発生する場合は，バージョンアップ後に次に示すオペランドを指定して，省略値が変更されないようにしてください。

- HiRDB Version 6 からバージョンアップした場合に，オペランドの省略値を変更しないとき（HiRDB Version 6 の省略値を使用するとき）は pd_sysdef_default_option オペランドに v6compatible を指定し

てください。

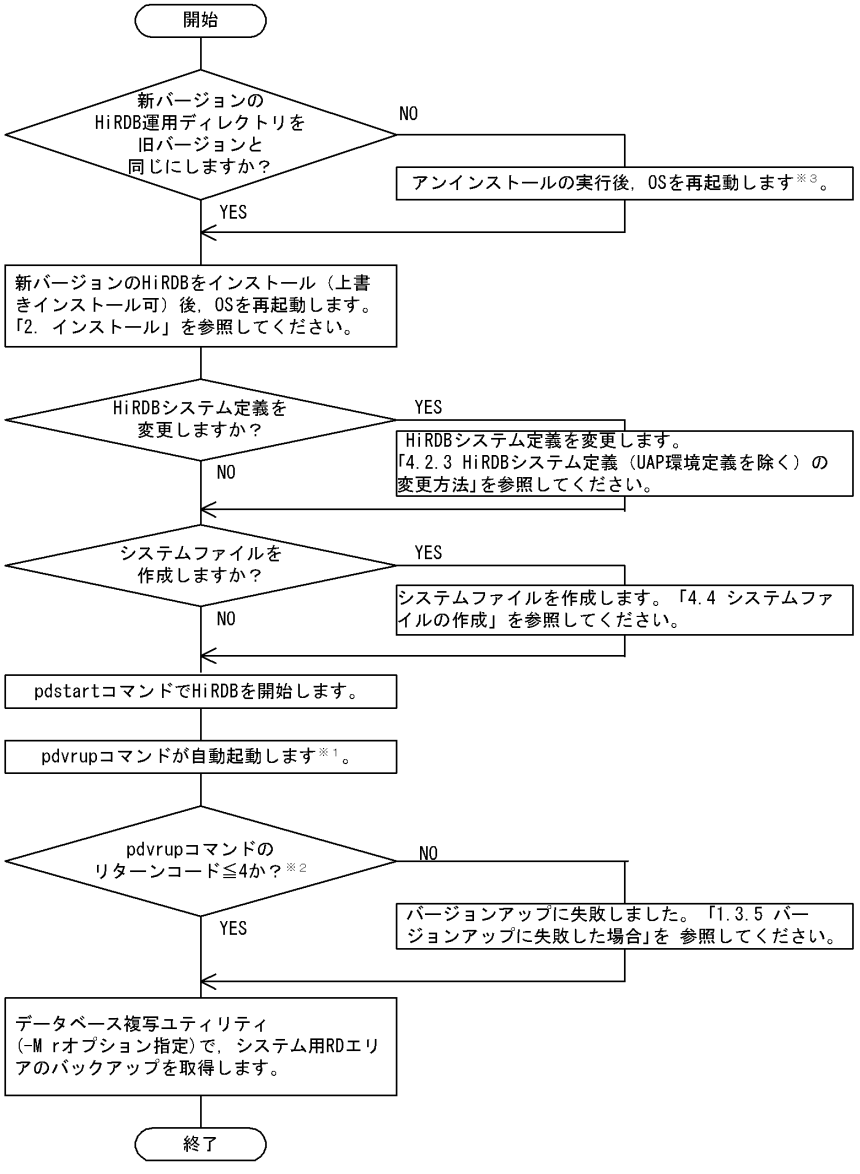
- HiRDB Version 7 からバージョンアップした場合に，オペランドの省略値を変更しないとき（HiRDB Version 7 の省略値を使用するとき）は pd_sysdef_default_option オペランドに v7compatible を指定してください。

省略値が変更になったオペランド，指定不要になったオペランド，及び pd_sysdef_default_option オペランドについては，マニュアル「HiRDB Version 8 システム定義」を参照してください。

### 1.3.2 旧バージョンと新バージョンを入れ替える場合

旧バージョンと新バージョンを入れ替える場合の操作手順を次の図に示します。

**図 1-2 旧バージョンと新バージョンを入れ替える場合の操作手順**




注※ 1

- システム共通定義で pd_auto_vrup = N を指定すると，pdvrup コマンドは自動起動しません。この場合，KFPS05203-Q メッセージ（pdvrup コマンドの入力要求メッセージ）が出力されたら，HiRDB 管理者が pdvrup コマンドを入力してください。
- 修正版 HiRDB への入れ替えの場合，pdvrup コマンドは自動起動しません。次の手順に進んでください。

注※ 2

pdvrup コマンドの実行結果は，メッセージログファイル又はイベントログにある KFPX24404-I メッ

セージを検索して確認してください。

注※ 3

HiRDB を旧バージョンに戻さないといけなくなった場合，旧バージョンのインストールディレクトリにインストールする必要があります。旧バージョンのインストールディレクトリを控えておいてください。

### 1.3.3 HiRDB のプラグインをバージョンアップする場合

HiRDB をバージョンアップするときは，プラグインもバージョンアップする必要がある場合があります。プラグインの前提になる HiRDB のバージョンとプラグインのバージョンアップの手順については，該当するプラグインのマニュアル及び「5.2　プラグインのバージョンアップ」を参照してください。

### 1.3.4 Java ストアドプロシジャ及び Java ストアドファンクションを使用する場合

HiRDB では，バージョン 07-02 以前は，Java ストアドプロシジャ及び Java ストアドファンクションを使用するために必要な JRE（Java 実行環境）を同梱していましたが，07-03 以降は同梱していません。07-02 以前の HiRDB を 07-03 以降にバージョンアップする場合，07-02 以前の HiRDB をインストールする時にインストールした JRE は，次の時点で削除されます。

- HiRDB のアンインストール時

HiRDB 07-03 以降にバージョンアップした場合の注意事項を次に示します。

- Java ストアドプロシジャ及び Java ストアドファンクションを使用する場合，使用前に JRE を入手しておく必要があります（各プラットフォームのベンダのホームページから JRE に関する情報を入手できます)。Java ストアドプロシジャ及び Java ストアドファンクションを使用するために必要な JRE のバージョンについては，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。
- Java ストアドプロシジャ及び Java ストアドファンクションを使用する場合，使用する JRE のルートディレクトリを pd_java_runtimepath オペランドに指定する必要があります。また，必要に応じて Java 仮想マシンのライブラリが格納されているディレクトリを pd_java_libpath オペランドに指定します。pd_java_runtimepath オペランド及び pd_java_libpath オペランドについては，マニュアル「HiRDB Version 8 システム定義」を参照してください。
- 新バージョンにバージョンアップした後も，旧バージョンでインストールした JRE を使用したい場合，JRE が削除される前にインストールディレクトリ及び HiRDB 運用ディレクトリ以外のディレクトリに退避してください。退避した JRE のディレクトリを pd_java_runtimepath オペランドに指定することで，退避した JRE で Java ストアドプロシジャ及び Java ストアドファンクションを使用できます。

### 1.3.5 バージョンアップに失敗した場合

ここでは，次に示す現象が発生した場合の対処方法について説明します。

- pdvrup コマンドを入力して 5 以上のリターンコードが返ってきた
- HiRDB 開始時にリターンコードが 5 以上の KFPX24404-I メッセージが出力された

この場合，一緒に出力されたメッセージを参照して対策してください。

### (1) HiRDB を終了させなくてもよいとき

失敗の原因を取り除いた後に，再度 pdvrup コマンドを入力してください。

### (2) HiRDB を終了させないといけないとき

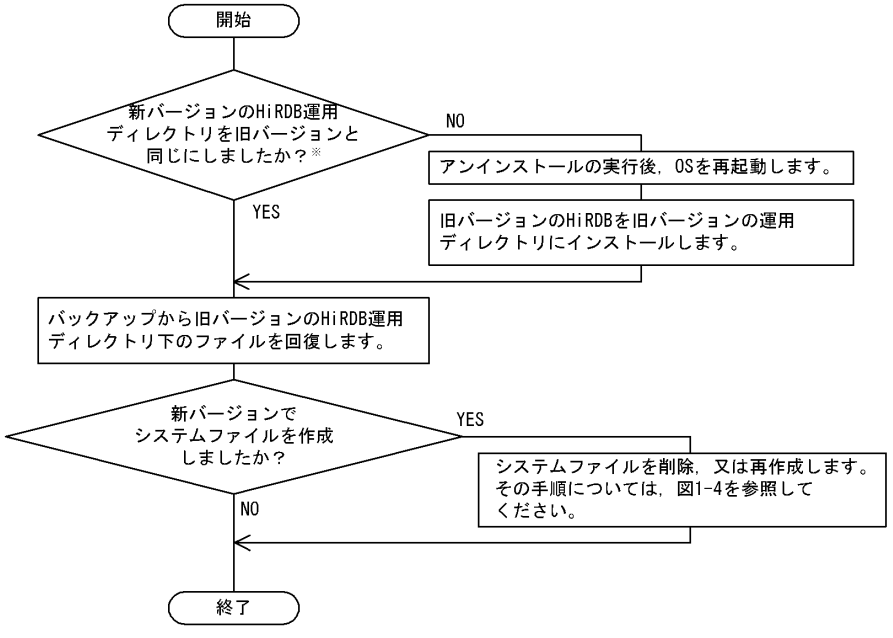
HiRDB を終了しないと，失敗の原因が取り除けない場合は，いったん HiRDB を終了してください。そして，失敗の原因を取り除いた後に，pdstart コマンドで HiRDB を開始してください。開始すると，pdvrup コマンドの入力要求メッセージ（KFPS05203-Q）が出力されるので，再度 pdvrup コマンドを入力してください。

### (3) HiRDB を旧バージョンに戻さないといけないとき

失敗の原因によっては，HiRDB を旧バージョンに戻して対処する必要があります。例えば，データディクショナリ用 RD エリアの容量が不足していてバージョンアップに失敗した場合は，HiRDB を旧バージョンに戻してデータベース構成変更ユティリティ（pdmod）で対策する必要があります。このような場合は，いったん HiRDB を旧バージョンに戻して，失敗の原因を取り除き，その後で再度バージョンアップをしてください。

HiRDB を旧バージョンに戻す手順を次の図に示します。

**図 1-3 HiRDB を旧バージョンに戻す手順（バージョンアップに失敗した場合）**



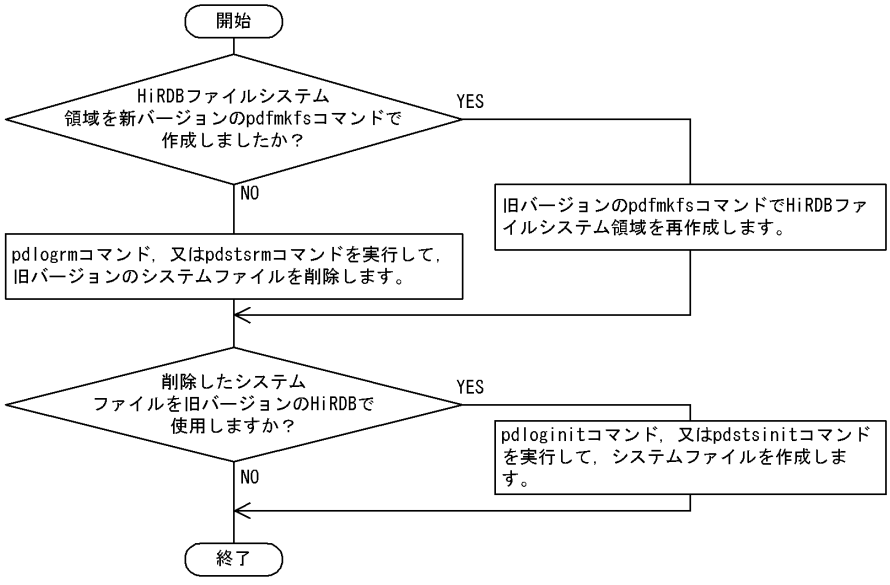

注※

バージョン 08-02 以降から，バージョン 08-02 より前の HiRDB に戻す場合は，NO へ進みます。このとき，新バージョンの HiRDB をアンインストールしないで旧バージョンの HiRDB をインストールした場合は，新バージョンがインストールしたレジストリの情報，ディレクトリ，及びファイルが残ってしまいます。これらが残ったままでも旧バージョンの HiRDB は動作しますが，［コントロールパネル］－［アプリケーションの追加と削除］の「現在インストールされているプログラム」に，同じセットアップ識別子の HiRDB が二つ表示されます。その場合は，「現在インストールされているプ

ログラム」で，両方のプログラムを削除した後，OS を再起動し，旧バージョンの HiRDB をインストールし直してください。

新バージョンで作成したシステムファイルの削除，又は再作成する手順を次の図に示します。

**図 1-4　新バージョンで作成したシステムファイルの削除又は再作成手順**




## 1.3.6　HiRDB を旧バージョンに戻す場合

HiRDB のバージョンアップに成功した後にバージョンダウンを行う場合（例えば，テストなどで一度 HiRDB をバージョンアップした後に，バージョンダウンして元の運用に戻したい場合など）は，旧バージョンで再構築する必要があります。

### (1) 前提条件

旧バージョンで再構築する場合には，バージョンアップ前にすべての RD エリアのバックアップ，及びシステム定義ファイルのバックアップを取得しておく必要があります。バックアップは，データベース複写ユティリティ（pdcopy）で取得します。

### (2) 再構築の手順

再構築の手順は，最初に HiRDB をインストール及び環境設定する手順と基本的に同じです。旧バージョンで再構築する場合の手順を次に示します。

1. HiRDB を停止します（pdstop コマンド）。
2. 旧バージョンをインストールします。
3. 旧バージョンの環境設定を行います。
   HiRDB ファイルシステム領域を作成します（pdfmkfs コマンドを実行）。※
   HiRDB ファイルシステム領域に，システムファイル（システムログファイル，シンクポイントダンプファイル，ステータスファイル）を作成します（pdloginit コマンド，及び pdstsinit コマンドを実行）。
   なお，新バージョンの HiRDB でシステム定義を変更した場合，バージョンアップ前に取得した旧バージョンのシステム定義ファイルに変更する（旧バージョンの HiRDB のシステム定義に戻す）必要があ

ります。
4. データベース回復ユティリティ（pdrstr）を起動するため，pdstart -r コマンドで HiRDB を開始します。
5. データベース回復ユティリティ（pdrstr）で，バージョンアップ前に取得したバックアップファイルからデータベースを回復します（全 RD エリアのリストア）。このとき，バージョンアップ後の HiRDB で更新したログを含むアンロードログファイルは使用しないでください。
6. HiRDB を停止します（pdstop コマンド）。
7. HiRDB を開始します（pdstart コマンド）。

注※

新バージョンの HiRDB で pdfmkfs コマンドを実行していない場合，旧バージョンでの HiRDB ファイルシステム領域の作成は不要です。

### (3) 注意事項

旧バージョンに戻す場合の注意事項を次に示します。

1. 系切り替え機能を使用している場合
   現用系，予備系ともに旧バージョンに戻してください。
2. 付加 P.P. をセットアップしている場合
   旧バージョンに戻すことで，付加 P.P. の前提バージョンを満たさなくなった場合，旧バージョンの HiRDB に対応した付加 P.P. に変更してください。
3. セキュリティ監査機能を使用している場合
   再構築の手順 3. で，システムファイルを作成するとき，監査証跡ファイルも作成してください。
4. HiRDB Datareplicator とのデータ連動機能を適用している場合
   - 再構築の手順 1. ～ 2. の間に HiRDB Datareplicator も同時に旧バージョンに戻す必要があります。
   - 再構築の手順 7. の後に HiRDB，HiRDB Datareplicator 連動環境を再初期化する必要があります。

# 1.4 修正版 HiRDB への入れ替え

修正版 HiRDB とは，07-02-/A のように，稼働中の HiRDB とバージョン番号及びリビジョン番号が同じで，「-mn」のコードがあるものです（下線部がコード）。07-03 より前のバージョンの場合，m は /，英字（I，O，P ～ T を除く），又は数字，n は A ～ Z のアルファベットです。07-03 以降は m，n は共に数字です。

修正版 HiRDB への入れ替えは，既存の HiRDB を終了しないで，稼働中に実行できます。ただし，MSCS 又は MSFC を導入して系切り替え機能を使用する場合に，MSCS 又は MSFC の汎用サービスに HiRDB のサービスを登録していると，HiRDB の稼働中に入れ替えることはできません。

## 1.4.1 修正版 HiRDB への入れ替え方法

修正版 HiRDB への入れ替えには，次の方法があります。

- インストーラによる入れ替え
- 修正パッチを Web から入手して適用

それぞれについて説明します。

### （1） インストーラによる入れ替え

インストーラを使用して修正版 HiRDB と入れ替えます。入れ替え方法には次の二つがあります。

#### (a) HiRDB を終了して入れ替え

HiRDB を終了して行う入れ替えはバージョンアップと同じ方法です。詳細は，「1.3　HiRDB のバージョンアップ」を参照してください。ただし，修正版への入れ替えの場合は次の点が異なります。

- 「1.3.1　バージョンアップ前にすること」の次の作業は不要です。
  - (1) 空き領域の確認
  - (3) メモリ容量の確認
- pdvrup コマンドの実行は不要です。

#### (b) HiRDB の稼働中に入れ替え

1.4.2 以降を参照してください。

### （2） 修正パッチを Web から入手して適用

稼働中の HiRDB と，バージョン及びリビジョン番号が同じ場合，修正パッチを適用することで最新版にできます。修正パッチは，Web からダウンロードして入手できます。入れ替え方法には次の二つがあります。

#### (a) HiRDB を終了して入れ替え

適用手順については，修正パッチに添付されている RELEASE.TXT を参照してください。

ただし，HiRDB の場合，同梱している製品があるため，パッチを適用する製品を選択する画面が表示されます。次の画面で選択した製品の修正パッチを適用します。

**(b) HiRDB の稼働中に入れ替え**

1.4.2 以降を参照してください。

## 1.4.2 前提条件

HiRDB の稼働中に入れ替えができるのは，次の条件を満たしている場合です。

● バージョン，HiRDB サーバの種別，アドレッシングモード
修正版 HiRDB と稼働中の HiRDB とで次に示す項目が同じである必要があります。なお，次の項目は pdadmvr コマンドで確認できます。
- バージョン番号，リビジョン番号
- HiRDB サーバの種別（HiRDB/ シングルサーバか，HiRDB/ パラレルサーバか）
- アドレッシングモード（32 ビットモードか，64 ビットモードか）

● ディスクの空き容量
HiRDB 運用ディレクトリに，現在稼働中の HiRDB と修正版 HiRDB の両方が格納できる程度のディスクの空き容量が必要です。修正版 HiRDB を格納するのに必要な空き容量はリリースノートを参照してください。このほかに，修正版 HiRDB を新規にインストールをする必要があり，そのためのディスク容量も必要となります。

● HiRDB クライアント
オンライン業務をしている HiRDB クライアントは，入れ替えをする HiRDB サーバ以外で稼働している必要があります。HiRDB クライアントが入れ替えをする HiRDB サーバ上で稼働している場合，HiRDB クライアントを停止し，オンライン業務を停止する必要があります。

● クライアントライブラリ
オンライン業務をしている HiRDB クライアントが利用している HiRDB/Developer's Kit 及び HiRDB/Run Time のバージョンは 07-00 以降である必要があります。07-00 より前のバージョンを使用している場合，入れ替えの途中で接続中の HiRDB クライアントとの接続が切断されます。

● 自動再接続機能の適用
HiRDB に接続する HiRDB クライアントは，自動再接続機能（PDAUTORECONNECT=YES）を使用する必要があります。自動再接続機能を使用していない場合，入れ替えの途中で接続中のクライアントとの接続が切断されます。自動再接続機能については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

### 1.4.3 入れ替え前にすること

HiRDB/ パラレルサーバの場合，修正版 HiRDB への入れ替えをする前に OS 環境ファイルを設定してください。

全ユニットのサーバマシンで SERVICES ファイル（%windir%¥system32¥drivers¥etc¥SERVICES）に連絡用ポートのサービス名（pdrshsrv-RNW）と，リモートコマンドシェルの接続用ポート番号を設定します。設定するポート番号は全ユニットのサーバマシンで同一とします。また，マシン内では同じ番号は使用しないでください。

### 1.4.4 入れ替えの手順

修正版 HiRDB への入れ替え手順を次に示します。

1. **修正版 HiRDB のインストール**

   修正版 HiRDB をインストールする場合の注意事項を次に示します。

   - 修正版 HiRDB は，上書きインストールではなく，新規にインストールしてください。
   - ［プログラムメンテナンス用セットアップ］でインストールした HiRDB は入れ替えのためだけに利用できます。
   - HiRDB/ パラレルサーバの場合，全ユニットで同一パス名のディレクトリに修正版 HiRDB をインストールしてください。

   インストールする修正版 HiRDB の媒体によって手順が異なります。媒体ごとに手順を示します。

   **インストーラ（HiRDB インストール用 CD-ROM）の場合（HiRDB 07-01 以前）**

   次の手順でインストールしてください。

   1. 修正版 HiRDB のインストール用 CD-ROM 中の 01_srv¥Setup.exe をダブルクリックし，インストーラを起動します。
   2. インストーラ実行中に表示される［セットアップ方法］画面で［プログラムメンテナンス用セットアップ］を選択して，稼働中の HiRDB とは異なるディレクトリに修正版 HiRDB をインストールします。

   **インストーラ（統合 CD-ROM）の場合（HiRDB 07-02 以降）**

   1. 修正版 HiRDB の統合 CD-ROM 中の hcd_inst.exe を実行して，日立総合インストーラを起動します。
   2. ［日立総合インストーラ］画面で，HiRDB/ シングルサーバの場合は［HiRDB/Single Server］を，HiRDB/ パラレルサーバの場合は［HiRDB/Parallel Server］を選択して，HiRDB のインストーラを起動します。
   3. HiRDB のインストーラの［プログラムプロダクトの選択］画面で，HiRDB/ シングルサーバの場合は［HiRDB/Single Server］を，HiRDB/ パラレルサーバの場合は［HiRDB/Parallel Server］を選択して，HiRDB/Single Server 又は HiRDB/Parallel Server のインストーラを起動します。
   4. HiRDB/Single Server 又は HiRDB/Parallel Server のインストーラ実行中に表示される［セットアップ方法］画面で［プログラムメンテナンス用セットアップ］を選択して，稼働中の HiRDB とは異なるディレクトリに修正版 HiRDB をインストールします。

   **修正パッチ（HiRDB 07-03 以降）の場合**

   1. 稼働中の HiRDB をインストールした時に使用した統合 CD-ROM 中の hcd_inst.exe を実行して，日立総合インストーラを起動します。
   2. 「インストーラ（統合 CD-ROM）の場合（HiRDB 07-02 以降）」の手順 2 ～ 4 と同じ操作をして，HiRDB をインストールします。

3. 修正版 HiRDB の修正パッチを実行して，2. でインストールした HiRDB にパッチを適用します。適用手順については，修正パッチに添付している RELEASE.TXT を参照してください。なお，最新版の修正パッチには過去の修正内容が含まれているため，過去に発行された修正パッチを適用する必要はありません。

2. **修正版 HiRDB サービスの開始（HiRDB/ パラレルサーバの場合だけ）**
次の手順で修正版 HiRDB のサービスを開始してください。
1. ［コントロールパネル］の［サービス］をダブルクリックしてください。
2. ［サービス］リストボックスから修正版 HiRDB のサービス「HiRDB/ParallelServer-RNW」を選択して，［スタートアップ］ボタンをクリックしてください。
3. ［ログオン］がシステムアカウントに設定されているので，アカウントを選択してください。アカウントの右にあるボタンをクリックして，［ユーザーの追加］画面を表示させます。
4. ［名前の一覧］から HiRDB 管理者を選択します。［追加］をクリックし，［OK］をクリックしてください。
5. ［パスワード］画面で HiRDB 管理者のパスワードを入力してください。同じパスワードを［パスワードの確認入力］画面にも入力して，［OK］をクリックしてください。
6. ［サービス］リストボックスから修正版 HiRDB のサービスを選択して，［開始］をクリックしてください。

3. **入れ替え用ディレクトリに修正版 HiRDB をコピー**
［スタート］－［プログラム］－［HiRDBSingleServer-RNW］又は［HiRDBParallelServer-RNW］－［HiRDB コマンドプロンプト］を選択して，修正版 HiRDB の作業用コンソールを起動します。この作業用コンソールで，"pdprgcopy **稼働中の** HiRDB **運用ディレクトリ** " コマンドを実行して，修正版 HiRDB を稼働中の HiRDB 運用ディレクトリ下の入れ替え用ディレクトリ（%PDDIR%¥renew）にコピーしてください。HiRDB/ パラレルサーバの場合はシステムマネジャがあるユニットで作業用コンソールを起動して pdprgcopy コマンドを実行してください。

4. **入れ替え対象の HiRDB がオンライン状態であるかどうかの確認**
［スタート］－［プログラム］－［HiRDBSigleServer］又は［HiRDBParallelServer］－［HiRDB コマンドプロンプト］（セットアップ識別子を指定してインストールした場合は，［HiRDBSigleServer セットアップ識別子］又は［HiRDBParallelServer セットアップ識別子］になります）をクリックして，稼働中の HiRDB の作業用コンソールを起動し，pdls コマンドで全ユニットが「ACTIVE」と表示されていることを確認してください。確認が終了したら，HiRDB コマンドプロンプトは終了してください。

5. **ほかのプログラムの終了**
稼働中の HiRDB 運用ディレクトリにアクセスしているプログラム（エクスプローラ，コマンドプロンプトなど）や，レジストリにアクセスしているプログラム（regedit など），HiRDB 運用コマンドはすべて終了してください。

6. **修正版 HiRDB への入れ替え**
修正版 HiRDB の作業用コンソール上で "pdprgrenew **稼働中の** HiRDB **運用ディレクトリ** " コマンドを実行して，HiRDB の入れ替えをします。このコマンドを実行すると，稼働中の HiRDB をバックアップ用ディレクトリ（%PDDIR%¥renew_bak）に退避した後，稼働中の HiRDB を 3. でコピーした入れ替え用ディレクトリの修正版 HiRDB と入れ替えます。HiRDB/ パラレルサーバの場合はシステムマネジャがあるユニットの作業用コンソール上で pdprgrenew コマンドを実行してください。

7. **修正版 HiRDB サービスの停止（HiRDB/ パラレルサーバの場合だけ）**
［コントロールパネル］－［サービス］を選択して，修正版 HiRDB のサービス「HiRDB/ ParallelServer-RNW」を停止してください。

8. **修正版 HiRDB のアンインストール**
［スタート］－［プログラム］－［HiRDBSingleServer-RNW］又は［HiRDBParallelServer-RNW］－［HiRDBSingleServer のアンインストール］又は［HiRDBParallelServer のアンインストール］を選択して，1. でインストールした修正版 HiRDB をアンインストールしてください。なお，アンインストール時に修正版 HiRDB の運用ディレクトリ下のファイル及びディレクトリの一部が削除されなかった場合，手動で運用ディレクトリ下のファイル及びディレクトリを削除してください。

9. **OS 環境ファイルの設定（HiRDB/ パラレルサーバの場合だけ）**
「1.4.3　入れ替え前にすること」で設定した連絡用ポートのサービス名とポート番号を SERVICES ファイル（%windir%¥system32¥drivers¥etc¥SERVICES）から削除してください。

## 1.4.5　系切り替え機能使用時の入れ替え手順

系切り替え機能を使用している場合，HiRDB の稼働中に入れ替えができるのは次の場合です。

- スタンバイ型系切り替えの場合
実行系が現用系で稼働中のときだけです。実行系が予備系で稼働中のときは，系を切り替えてコマンドを実行してください。
- スタンバイレス型系切り替えの場合
すべての正規 BES が稼働中のときだけです。代替中は入れ替えできません。

系切り替え機能使用時の修正版 HiRDB への入れ替え手順を次に示します。

**● サーバモードで運用している場合**

**スタンバイ型系切り替え**

1. 予備系が実行系の場合，現用系が実行系になるように系を切り替えてください。
2. 待機系の HiRDB を停止してください。
3. 実行系で修正版 HiRDB への入れ替えを実行してください。
4. 待機系に修正版 HiRDB を上書きインストールしてください。
5. 1. で停止させた待機系の HiRDB を再起動します。待機系の HiRDB で pdstart コマンド（HiRDB/ パラレルサーバの場合は pdstart -q コマンド）を実行してください。

**1：1 スタンバイレス型系切り替え**

1. 代替 BES が稼働中の場合，系を切り戻してください。
2. 代替 BES ユニットの代替部の待機状態を解除してください。解除方法については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。
3. 実行系の正規 BES ユニットで修正版 HiRDB への入れ替えを実行してください。
2. で待機状態を解除した代替部は pdprgrenew コマンドを実行すると自動的に待機状態になるため，これ以降の操作は不要です。

**影響分散スタンバイレス型系切り替え**

1. 受け入れユニットのゲスト BES が稼働中の場合，正規ユニットに系を切り戻してください。
2. HA グループに属する，すべての稼働していないゲスト BES の受け入れ可能状態を解除してください。
3. 実行系の正規ユニットで修正版 HiRDB への入れ替えを実行してください。
2. で受け入れ可能状態を解除したゲスト BES は，pdprgrenew コマンドを実行すると自動的に受け入れ可能状態になるため，これ以降の操作は不要です。

**● モニタモードで運用している場合**

1. 予備系が実行系の場合，現用系が実行系になるように系を切り替えてください。
2. 実行系で修正版 HiRDB への入れ替えを実行してください。

3. 待機系に修正版 HiRDB を上書きインストールしてください。

### 1.4.6 注意事項

● 修正版 HiRDB へ入れ替えの実行不可
HiRDB の稼働状況によっては，修正版 HiRDB の入れ替えが実行できないことがあります。詳細については，マニュアル「HiRDB Version 8 コマンドリファレンス」の pdprgrenew コマンドの説明を参照してください。

● UAP のレスポンス遅延
pdprgrenew コマンドを実行している時間は，UAP のレスポンスが遅くなります。このため，比較的トラフィックが低い時間帯に実行することをお勧めします。

● 定義の変更
修正版 HiRDB への入れ替えに伴って，必要となるメモリサイズが変わり，システム定義の変更が必要となる場合があります。その場合は事前にシステム構成変更コマンド（pdchgconf コマンド）で HiRDB システムの定義を変更する必要があります。システム構成変更コマンドの使用方法については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

● 運用コマンド，ユティリティの実行
pdprgrenew コマンド実行中は運用コマンドやユティリティを実行しないでください。実行すると，HiRDB が停止している旨のエラーが表示されたり，HiRDB の入れ替え作業が失敗することがあります。

● 系切り替え機能使用不可
修正版への入れ替え中は系切り替え機能は使用できません。

● ホールダブルカーソルの無効
修正版への入れ替えではカーソル保持ができないので，ホールダブルカーソルを使用する UAP は入れ替え前後で実行できません。そのため，UAP はエラーとなります。

● UNTIL DISCONNECT 指定の LOCK 文が無効
修正版への入れ替えでは UNTIL DISCONNECT 指定の排他を保持できないため，UNTIL DISCONNECT 指定の LOCK 文を使用する UAP は修正版への入れ替え前後で実行できません。そのため，UAP はエラーとなります。

### 1.4.7 運用上の注意事項

修正版 HiRDB へ入れ替えるときの運用上の注意事項を次に示します。

- データベース構成変更ユティリティ（pdmod）で割り当てたグローバルバッファが無効になります。そのため，入れ替え後に再度グローバルバッファを割り当てる必要があります。
- pd_spool_cleanup_interval オペランドの時間のカウント開始時点が入れ替え時の時刻にリセットされます。
- pd_spool_cleanup オペランドに normal，又は force を指定している場合，入れ替え時に出力済みのトラブルシュート情報は削除されます。
- pdstbegin コマンドや pdstend コマンドを使用して，統計情報の取得条件を pd_statistics オペランドや pdstbegin オペランドの指定と異なる値に変更していた場合，次のようになります。
  - pd_statistics オペランドや pdstbegin オペランドを指定しないで起動した環境で pdstbegin コマンドで統計情報を取得していた場合，入れ替え後には統計情報を取得しなくなります。そのため，再度 pdstbegin コマンドを実行する必要があります。
  - pd_statistics オペランドに Y を指定，又は pdstbegin オペランドを指定して起動した環境で pdstend

コマンドで統計情報取得を中止した場合や，pdstbegin コマンドを実行して取得する統計情報の種類を変更していた場合，入れ替え後にはシステム共通定義に記述した指定で統計情報を取得します。そのため，再度 pdstend コマンドや pdstbegin コマンドを実行する必要があります。

- 絞込み検索で使用しているリストがなくなるため，入れ替え後に ASSIGN LIST 文でリストを再作成する必要があります。
- pdchprc コマンドで変更した常駐プロセス数は HiRDB システム定義で指定した常駐プロセス数に戻るため，入れ替え後に再度 pdchprc コマンドを実行する必要があります。
- 修正版 HiRDB への入れ替えをすると，システムログファイルが切り替わります。入れ替え前に，スワップできるシステムログファイルがあることを確認し，不足している場合は次の対処をしてください。
  - スワップできるシステムログファイルがない場合
    アンロード待ち状態のファイルがあれば，アンロードしてください。アンロード待ち状態のファイルがなければ，システム構成変更コマンド（pdchgconf コマンド）を使用してスワップできるログファイルを追加してください。システム構成変更コマンドの使用方法については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。
    スワップできるシステムログファイルがない状態で入れ替えを実行すると，KFPS01256-E メッセージを出力後，Psjnf07 又は Psjn381 のコードでアボートして HiRDB が停止します。この場合は，スワップできるファイルを準備してから pdstart コマンドで HiRDB を起動してください。
  - スワップできるシステムログファイルが一つだけの場合
    修正版 HiRDB への入れ替えはできますが，入れ替え中にスワップできるファイルがないことを示す KFPS01224-I メッセージが出力されます。入れ替え後にスワップできるシステムログファイルを準備してください。
- 修正版 HiRDB への入れ替えをすると，メッセージログファイルが切り替わります。ただし，メッセージログファイルの切り替えを知らせるメッセージ（KFPS01910-I など）は表示されません。メッセージログファイル中のメッセージを保存したい場合は，入れ替え前にメッセージログファイルをバックアップしてください。
- 修正版 HiRDB への入れ替えをすると，入れ替え対象の HiRDB のサービスや，MSCS 又は MSFC のリソースが一時的に停止，又はオフラインになります。

## 1.4.8 関連製品の制限及び注意事項

● プラグイン
プラグインを利用している場合も HiRDB の稼働中に修正版 HiRDB へ入れ替えができます。ただし，プラグインの入れ替えはできません。

● HiRDB Datareplicator 連携機能
HiRDB Datareplicator を使用してデータ抽出中の HiRDB に対して pdprgrenew コマンドを実行しないでください。オンライン業務を停止しないで修正版 HiRDB への入れ替えをする場合，抽出側の HiRDB Datareplicator を終了させる必要があります。ただし，HiRDB Datareplicator 連携は中止させないでください（pdrplstop コマンドは実行しないでください）。HiRDB Datareplicator 連携を中止させると，抽出側 DB と反映側 DB が不整合になることがあります。

## 1.4.9 障害時の運用

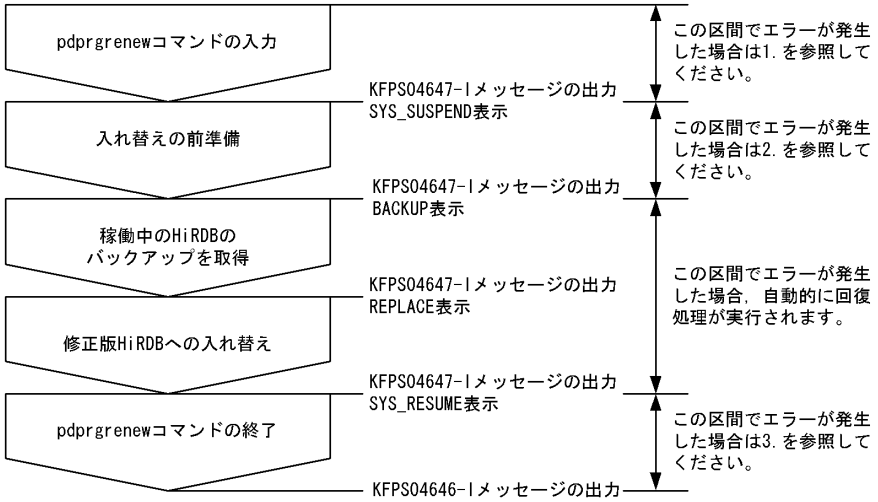
### (1) エラー発生時の対処

修正版 HiRDB への入れ替え実行中にエラーが発生した場合，pdprgrenew コマンドは自動的に入れ替え前の HiRDB に戻して HiRDB を動作させようとします。エラー発生後に，コマンドがリターンコード 12

の KFPS04646-I メッセージを出力して終了した場合，HiRDB を入れ替え前に戻す作業が失敗しています。そのため，標準エラー出力やイベントログに出力されたエラーメッセージと KFPS04647-I メッセージを参照して対処してください。

修正版 HiRDB への入れ替え中にエラーが発生した場合の対処方法を次の図に示します。

**図 1-5 修正版 HiRDB への入れ替え中にエラーが発生した場合の対処方法**




1. pdprgrenew コマンドがエラーになった要因を取り除いて，pdprgrenew コマンドを再度実行してください。
2. HiRDB のサーバプロセスがあれば，pdstop -f コマンドで HiRDB を強制終了してから pdprgrenew -b コマンドを実行してください。HiRDB のサーバプロセスがなければ，pdprgrenew -b コマンドを実行してください。pdprgrenew -b コマンドを実行すると，回復処理として，入れ替え前の HiRDB で再開始しようとします。
   また，HiRDB の終了処理失敗に関するエラーメッセージ及びアボートコードが表示されることがあります。メッセージの対処方法に従って，入れ替え前の HiRDB の稼働環境を確認し，対処してください。
3. HiRDB のサーバプロセスがあれば，pdstop -f コマンドで HiRDB を強制終了してから pdprgrenew -b コマンドを実行してください。HiRDB のサーバプロセスがなければ，pdprgrenew -b コマンドを実行してください。pdprgrenew -b コマンドを実行すると，回復処理として，修正版 HiRDB を入れ替え用ディレクトリに戻します。
   また，HiRDB の開始処理失敗に関するエラーメッセージ及びアボートコードが表示されることがあります。入れ替え後の修正版 HiRDB が動作するための環境に問題がある可能性があるので，表示されるメッセージに従って対処してください。

### (2) 入れ替え失敗時に HiRDB が入れ替え前の状態に戻っているかどうかの確認

修正版 HiRDB への入れ替えに失敗したとき，入れ替え前の HiRDB に戻っているかどうかは，次の項目をチェックして確認できます。次の項目を満たしていれば，入れ替え前の HiRDB に戻っています。

- pdadmvr -s コマンドで表示されるバージョンが入れ替え前の HiRDB バージョンと一致する
- HiRDB がオンライン状態（pdls コマンドの結果，全ユニットが「ACTIVE」と表示）である
- バックアップ用ディレクトリ（%PDDIR%¥renew_bak）がない

# 2 インストール

この章では，インストール前に必要な作業，HiRDB のインストール手順，付加 PP インストール時の注意，及び HiRDB のアンインストール手順について説明します。
また，Windows ファイアウォールの例外リストへの登録方法と削除方法についても説明します。


2.1 **インストール前に必要な作業**

2.2 HiRDB **のインストール手順**

2.3 **インストール後の作業**

2.4 **付加** PP **のインストール時の注意**

2.5 HiRDB **のアンインストール手順**

2.6 Windows **ファイアウォールの例外リストへの登録**

2.7 Windows **ファイアウォールの例外リストからの削除**

# 2.1 インストール前に必要な作業

ここでは，HiRDB をインストールする前に必要な作業について説明します。説明する項目を次に示します。

- サーバマシン環境の確認
- HiRDB 管理者の登録
- OS 環境ファイルの設定
- ホスト名の登録

## 2.1.1 サーバマシン環境の確認

HiRDB を Windows にインストールする前に，使用するサーバマシンの環境を確認してください。

### (1) 適用機種の確認

- HiRDB は Windows の PC/AT 互換機版しか対応していません。PC/AT 互換機版の Windows であることを確認してください。
- HiRDB/ パラレルサーバの場合，すべてのサーバマシンで同じバージョンの Windows を使用してください。
- HiRDB/ パラレルサーバをインストールする形態は単一ドメインの形態をお勧めします。

### (2) システム環境変数 TZ の確認

サーバマシンのシステム環境変数 TZ に JST-9 が設定されているかどうかを確認してください。システム環境変数 TZ の確認手順を次に示します。

1. ［コントロールパネル］の［システム］をダブルクリックしてください。
2. ［システムのプロパティ］の［環境］タブを選択してください。
3. システム環境変数の変数「TZ」が JST-9 になっているかどうかを確認してください。
4. TZ がない場合は TZ を追加してください。TZ の値が JST-9 でない場合は，JST-9 に変更してください。

**注意事項**

システム環境変数 TZ の値と，HiRDB システム定義の設定内容（システム共通定義の TZ オペランドの設定値）が同じである必要があります。TZ に JST-9 以外の値を設定する場合は，システム共通定義の TZ オペランドの指定値を変更する必要があります。

### (3) ディスク容量の確認

インストーラでもディスク容量をチェックしていますが，インストールする前にディスクの空き容量が十分かどうか確認してください。必要な空き容量の目安を次に示します。

**（インストール時に必要な容量）＋（HiRDB 動作時に自動的に確保する容量）**

インストール時に必要な容量：

- HiRDB/ シングルサーバの場合：93.4MB
- HiRDB/ パラレルサーバの場合：95.9MB

  なお，バージョンによってこの数値は変動する場合があります。リリースノートのメモリ所要量とディスク占有量を参照してください。

HiRDB 動作時に自動的に確保する容量：1MB ＋共用メモリサイズ

共用メモリサイズの見積もりについては，「15. HiRDB のメモリ所要量」を参照してください。

インストール時にエラーメッセージが出力されてエラーの理由に 'No-space'（書き込むファイルに十分な容量がありません）が出力された場合，次の原因が考えられます。

- ディスク容量が十分な状態でこのエラーになる場合，2048 メガバイト以上の HiRDB ファイルシステム領域を使用する指定（pd_large_file_use）をしていない可能性があります。

## (4) 仮想メモリの確認

サーバマシンの仮想メモリの容量（**実メモリ容量＋ページングファイルの容量の合計値**）は，**［コントロールパネル］**の**［システム］**をダブルクリックして表示される［システムのプロパティ］で確認できます。ページングファイルの容量は［システムのプロパティ］の**［パフォーマンス］**タブで確認及び変更できます。ページングファイルは，同じドライブに連続した領域を作成するように，初期サイズと最大サイズを同じ値（固定値）にしてください。ページングファイルを連続で使用できない場合，HiRDB がメモリ不足で異常終了します。

必要な仮想メモリの目安を表 2-1 及び表 2-2 に示します。また，実際に指定する値には，Windows やほかのプログラムが使用する容量を加えてください。仮想メモリを変更した場合には，必ず Windows を再起動してください。

**表 2-1　必要な仮想メモリの容量（HiRDB/ シングルサーバの場合）**

| 項目 | メモリ所要量（単位：MB） |
|---|---|
| HiRDB/ シングルサーバのシステムプロセス | 207.1 以上 |
| HiRDB/ シングルサーバの下で動作する同時実行サーバプロセスのプロセス固有領域 | 4.5 以上／ユーザ |

**仮想メモリ容量（MB）＝ {207.1 ＋（4.5 × n)} ＋ m**

n：HiRDB 下で動作する同時実行ユーザ数

m：共用メモリを HiRDB 運用ディレクトリ下のファイルに割り当てる場合は 0,
　ページングファイルに割り当てる場合は共用メモリサイズ

**表 2-2　必要な仮想メモリの容量（HiRDB/ パラレルサーバの場合）**

| 項目 | メモリ所要量（単位：MB） |
|---|---|
| HiRDB/ パラレルサーバのシステムプロセス | 292.5 以上 |
| HiRDB/ パラレルサーバの下で動作するサーバプロセスのプロセス固有領域 | 10.8 以上／フロントエンドサーバ<br>14.0 以上／ディクショナリサーバ<br>14.0 以上／バックエンドサーバ |
| HiRDB/ パラレルサーバの下で動作するユーザプロセスのプロセス固有領域 | 5.5 以上／フロントエンドサーバ<br>5.5 以上／ディクショナリサーバ<br>5.5 以上／バックエンドサーバ |

**仮想メモリ容量（MB）＝**
**{292.5 ＋（10.8 × x）＋（14.0 ×（y ＋ z））＋（5.5 ×（x ＋ y ＋ z））× n} ＋ m**

x：HiRDB 下で動作するフロントエンドサーバ数

y：HiRDB 下で動作するディクショナリサーバ数

z：HiRDB 下で動作するバックエンドサーバ数

n：HiRDB 下で動作する同時実行ユーザ数

m：共用メモリを HiRDB 運用ディレクトリ下のファイルに割り当てる場合は 0,
　ページングファイルに割り当てる場合は共用メモリサイズ

**仮想メモリの容量に関する注意事項**

- バージョンによって数値が変動する場合があります。リリースノートのメモリ所要量とディスク占有量を参照してください。
- 共用メモリを HiRDB 運用ディレクトリ下のファイルとページングファイルのどちらに割り当てるかは，pdntenv コマンドで指定します。指定方法については，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。
- HiRDB Datareplicator のメモリ所要量については，マニュアル「HiRDB データ連動機能 HiRDB Datareplicator Version 8」を参照してください。
- HiRDB Dataextractor のメモリ所要量については，マニュアル「データベース抽出・反映サービス機能 HiRDB Dataextractor Version 8」を参照してください。

### (5) システムキャッシュの確認

Windows で使用するファイルのシステムキャッシュは，設定方法によっては使用できるメモリを圧迫することがあります。システムキャッシュの設定を変更する手順を次に示します。

1. ［コントロールパネル］の［ネットワーク］をダブルクリックします。
2. ［ネットワーク］の［サービス］タブを選択します。
3. ［サーバー］を選択して，［プロパティ］をクリックします。
4. ［サーバー］のダイアログボックスの値が「ファイル共有のスループットを最大にする」に設定されていたら，［ネットワークアプリケーションのスループットを最大にする］を選択して［OK］ボタンをクリックします。
5. Windows を再起動します。

### (6) ファイルシステムの確認

- HiRDB が使用するファイルシステムに NTFS と FAT を使用できます。ただし，FAT についてはパス名の英字の全角大文字と小文字を区別しません。このため，同一名称がある場合は同一ファイルと認識するため，動作を保障できません。NTFS 及び FAT については，Windows のドキュメントを参照してください。
- HiRDB を NTFS にインストールする場合，HiRDB 運用ディレクトリ下のファイル（環境変数 PDDIR で指定したディレクトリ）を圧縮しないでください。このディレクトリを圧縮した場合，HiRDB は正常に動作しません。

### (7) リソース数に関連する環境変数の見積もり

一つのサーバマシン内で使用するリソース数を見積もってください。見積もり方法については，「20. リソース数に関連する環境変数の見積もり」を参照してください。

## 2.1.2 HiRDB 管理者の登録

Windows のシステム管理者（Administrators 権限を持つユーザ）を HiRDB 管理者としてください。HiRDB/ パラレルサーバの場合，又は IP アドレスを引き継がない系切り替え機能を使用する場合は各サーバマシンを同じシステム管理者にしてください。

HiRDB 管理者の登録は，プライマリ・ドメインコントローラ上の［管理ツール］－［ドメインユーザーマネージャ］，又は各マシンの［管理ツール］－［ユーザーマネージャ］で行います。

なお，HiRDB の予約語は認可識別子として使用しないでください。予約語については，次に示すマニュアルを参照してください。

- 「HiRDB Version 8 UAP 開発ガイド」
- 「HiRDB Version 8 SQL リファレンス」

## 2.1.3 OS 環境ファイルの設定

次に示す環境下ではリモートシェルを使用します。

- HiRDB/ パラレルサーバの場合
- IP アドレスを引き継がない系切り替え機能を使用する場合

そのため，コマンドを発行するクライアントマシンと，サーバマシンの SERVICES ファイル（%windir%¥system32¥drivers¥etc¥SERVICES）に，連絡用ポートのサービス名※とクライアントマシンで起動するリモートシェル接続用のポート番号が必要になります。設定するポート番号は HiRDB をインストールするそれぞれのマシンで同じにしてください。また，マシン内では同じ番号は使用しないでください。指定例を次に示します。

（例）

```
pdrshsrv    29999/tcp
```

- 連絡用ポートのサービス名として pdrshsrv を指定します。
- 任意のポート番号を指定します。省略値は 29999 です。

注※

マルチ HiRDB を使用して複数の HiRDB システムを構成する場合，HiRDB システム単位に連絡用ポートのサービス名を指定できます。指定方法については，「10.1　マルチ HiRDB のシステム設計」を参照してください。

**注意事項**

HiRDB/ シングルサーバで IP アドレスを引き継がない系切り替え機能を使用する場合，pdntenv コマンドの -ro オプションでリモート系コマンドを使用するように指定する必要があります。pdntenv コマンドについては，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

## 2.1.4 ホスト名の登録

HiRDB が使用するホスト名（システム定義及びクライアント環境定義で指定するホスト名）を hosts ファイル又は DNS などに登録し，名前解決してください。

システム定義及びクライアント環境定義中にホスト名を指定する場合，ホスト名，IP アドレス，又は FQDN のどれかの形式で指定します。

また，HiRDB/ シングルサーバだけで HiRDB システムが構成されている※場合，システム定義及びクライアント環境定義中にループバックアドレスを指定できます。ループバックアドレスを指定すると，ホスト名の登録が不要になります。

注※

HiRDB/ シングルサーバだけで HiRDB システムが構成されているとは，次に示す条件を満たすことをいいます。

- HiRDB クライアントと HiRDB サーバが同一マシンにある（HiRDB クライアントが別マシンにない）

**参考**

ループバックアドレスとは，127.0.0.0 ～ 127.255.255.255 の範囲の IP アドレス（例：127.0.0.1）のことです。
ループバックアドレスとして使用できる IP アドレスは OS の仕様に依存します。
また，HiRDB では，localhost を通常のホスト名として扱うため，システム定義などにホスト名として localhost を指定する場合はホスト名を登録し，名前解決しておく必要があります。

# 2.2 HiRDB のインストール手順

**実行者　HiRDB 管理者**

ここでは，HiRDB のインストール手順について説明します。ここで説明する項目を次に示します。

- インストール前の注意
- HiRDB サーバのインストール手順
- インストールが終わると設定される環境変数
- インストール後の注意
- インストール時に 1073 エラーになったときの対処

## 2.2.1 インストール前の注意

1. HiRDB のインストールは，Administrators 権限があるユーザが実行してください。Administrators 権限がないユーザがインストールを実行すると，エラーダイアログを表示してインストールを中止します。
2. 電源投入時にはネットワークドライブは接続されていないため，サービスを起動できません。このため，ネットワークドライブにはインストールしないでください。
3. アンインストール後にインストールをする場合は，アンインストールを実行した後に必ず Windows を再起動してください。再起動しないでインストールした場合，インストール終了直後に 1073 エラーが表示されます。このときの対処については，「2.2.5　インストール時に 1073 エラーになったときの対処」を参照してください。

## 2.2.2 HiRDB のインストール手順

**〈手順〉**

1. インストールする前に，すべての Windows アプリケーションを終了させてください。
2. 統合 CD-ROM 中の hcd_inst.exe を実行して，日立総合インストーラを起動してください。
3. ［日立総合インストーラ］画面で，HiRDB/ シングルサーバの場合は［HiRDB/Single Server］を，HiRDB/ パラレルサーバの場合は［HiRDB/Parallel Server］を選択して，［インストール実行］をクリックしてください。HiRDB のインストーラが起動します。
4. HiRDB のインストーラの［プログラムプロダクトの選択］画面で，HiRDB/ シングルサーバの場合は［HiRDB/Single Server］を，HiRDB/ パラレルサーバの場合は［HiRDB/Parallel Server］を選択して，［次へ］をクリックしてください。選択したプログラムプロダクトのインストーラが起動します。
5. ［セットアップ方法］画面でセットアップ方法を選択します。通常は［標準セットアップ］を選択します。マルチ HiRDB を使用する場合は，［識別子付きセットアップ］を選択します。マルチ HiRDB については，「10.1.1　マルチ HiRDB のインストール」を参照してください。
6. ［セットアップ識別子の設定］画面でセットアップ識別子を指定します。セットアップ識別子は 4 文字以内の半角英数字で指定します。なお，［標準セットアップ］を選択した場合は，［セットアップ識別子の設定］画面は表示されません。
7. ［ユーザの情報］画面で，ユーザの情報として名前と会社名を入力します。
8. ［インストール先の選択］画面で，インストールディレクトリを指定します。省略値を次に示します。
   なお，この章では，これ以降「Windows のインストール先ドライブ」を C: と仮定します。
   HiRDB/ **シングルサーバの場合：**Windows のインストール先ドライブ ¥win32app¥hitachi¥hirdb_s
   HiRDB/ **パラレルサーバの場合：**Windows のインストール先ドライブ ¥win32app¥hitachi¥hirdb_p

セットアップ識別子を指定してインストールした場合は，「hirdb_s」又は「hirdb_p」にセットアップ識別子が付いたディレクトリ名称となります。
**（例）**Windows **のインストール先ドライブ** ¥win32app¥hitachi¥hirdb_sUNT1
省略値で問題なければ，そのまま **［次へ］** をクリックしてください。
インストール先ディレクトリ名を変更する場合，次のことに注意してください。
・ドライブで始まり，次の要素で構成される文字列で指定してください。
英数字
_（下線）
.（ピリオド）
△（空白）
（（左括弧）
）（右括弧）
パス区切りの ¥
・ドライブだけの指定はできません。
・全角文字，及び特殊記号はディレクトリ名に使用できません。
・パス名の長さは 200 文字（バイト）以内にしてください。

9. Windows XP Service Pack 2 以降の場合，［Windows ファイアウォールへの登録］画面で，HiRDB サーバのプログラムを Windows ファイアウォールの例外リストに登録するかどうかを選択できます。Windows ファイアウォールを有効にしている環境で HiRDB を使用する場合は，例外リストに登録してください。
10. ［標準セットアップ］の場合，システムの環境変数に HiRDB の環境変数を設定するかどうかを選択してください（Windows Server 2003(IPF)，及び Windows (x64) の場合を除く）。HiRDB の環境変数については，「2.2.3　HiRDB の環境変数」を参照してください。
11. ［ファイルコピーの開始］画面で，設定した内容を確認して，問題がなければ［次へ］をクリックしてください。インストールが始まります。
12. インストールが終了すると，［セットアップの完了］画面が表示されます。
再起動してもよい場合は，［はい，直ちにコンピュータを再起動します。］を選択して，［完了］をクリックしてください。
ほかのプログラムプロダクトをインストールするなど，すぐに再起動しない場合は，［いいえ，後でコンピュータを再起動します。］を選択して，［完了］をクリックしてください。ただし，HiRDB を使う前に必ず Windows を再起動してください。

## 2.2.3　HiRDB の環境変数

### (1) 設定される環境変数

インストール時に，システムの環境変数に HiRDB の環境変数を設定する選択をした場合，インストールが終わると次の表に示す環境変数がシステム環境変数に設定されます。システム環境変数は［コントロールパネル］の［システム］をダブルクリックして，［システムのプロパティ］の［環境］タブで確認できます。

**表 2-3　システム環境変数に設定される環境変数**

| 環境変数 | 設定する内容 |
|---|---|
| PDDIR ※1 | HiRDB 運用ディレクトリ（インストールディレクトリ）が設定されます。<br>（例）C:¥win32app¥hitachi¥hirdb_s |
| PDCONFPATH ※1 | HiRDB システム定義ファイルを格納するディレクトリが設定されます。<br>（例）C:¥win32app¥hitachi¥hirdb_s¥CONF |

| 環境変数 | 設定する内容 |
|---|---|
| PATH ※2 | %PDDIR%¥CLIENT¥UTL 及び %PDDIR%¥BIN が追加されます。<br>（例）C:¥win32app¥hitachi¥hirdb_s¥CLIENT¥UTL<br>C:¥win32app¥hitachi¥hirdb_s¥BIN<br>なお，%PDDIR%¥BIN は，PATH の最後に追加されます。 |
| PDUXPLDIR | HiRDB の作業ディレクトリが設定されます。<br>（例）C:¥win32app¥hitachi¥hirdb_s¥UXPLDIR |

注※1

PDDIR の絶対パス名長は，200 バイト以内，PDCONFPATH の絶対パス名長は，213 バイト以内で設定してください。

注※2

PATH の設定値が 512 バイト以上の場合，表で示す環境変数が正しく設定されません。この場合，環境変数の設定値を確認して，必要ならば設定値を修正してください。

なお，LANG 環境変数は設定されません。そのため，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照して，ユーザが設定してください。

### (2) 注意事項

標準セットアップの場合，システム全体に影響を与えないように，HiRDB の環境変数を設定するかどうかを選択できます（Windows Server 2003 (IPF)，及び Windows (x64) の場合を除く）。HiRDB の環境変数の設定方法選択時の注意事項を次に示します。

**システム環境変数に HiRDB の環境変数を設定しなかった場合の注意事項**

- 運用方法がマルチ HiRDB と同じになります。マルチ HiRDB については，「10. マルチ HiRDB の設計」を参照してください。
- HiRDB の標準セットアップを前提とした製品を使用する場合，環境変数を設定する必要があります。詳細は，関連製品のリリースノート，又はマニュアルを参照してください。
- マルチ HiRDB に対応した関連製品でも，HiRDB の環境変数の設定する必要な場合があります。設定の必要な環境変数については，関連製品のリリースノート，又はマニュアルを参照してください。

**システムの環境変数に HiRDB の環境変数を設定した場合の注意事項**

ほかの製品が HiRDB の VC ランタイム（msvcr71.dll）を参照する場合があります。HiRDB が提供する VC ランタイムが，ほかの製品が提供する VC ランタイムより古い場合，ほかの製品が意図しない動作をするおそれがあります。そのため，HiRDB のディレクトリを PATH 環境変数の最後に設定する必要があります。HiRDB のインストール直後は，PATH 環境変数の最後に HiRDB のディレクトリを設定しますが，ほかの製品をインストールした場合などは，HiRDB のディレクトリが PATH 環境変数の最後にならなくなることがあるため，注意してください。

**そのほかの注意事項**

08-02 より前のバージョンでは，インストール時に PATH 環境変数に登録した HiRDB のインストールパスを削除していないため，PATH 環境変数に使用していない HiRDB のインストールパスが残っていることがあります。08-02 以降のバージョンでは，使用していない HiRDB のインストールパスは PATH 環境変数から削除されます。

## 2.2.4 インストール後の注意

1. HiRDB/ パラレルサーバの場合，又は IP アドレスを引き継がない系切り替え機能を使用する場合，次のように設定する必要があります。

- サービスのスタートアップ種別：自動
- サービス実行時に使用するログオンのアカウント：HiRDB 管理者

設定方法については OS のマニュアルを参照してください。

2. HiRDB をインストールした後には，必ず Windows を再起動してください。
3. HiRDB で使用する文字コードは，**シフト** JIS **漢字コード**（SJIS）が設定されています。

### 2.2.5 インストール時に 1073 エラーになったときの対処

アンインストール後にインストールをする場合は，アンインストールを実行した後に必ずリブートしてください。リブートをしないでインストールした場合，インストール終了直後に 1073 **エラー**が表示されます。ここでは，1073 エラーが表示されたときの対処方法を説明します。

1. Windows を再起動して，アンインストール時のレジストリ及びサービス登録の内容が反映されるようにします。
2. 通常と同じようにインストールして Windows を再起動します。インストール先はエラーになったときと同じディレクトリに指定してください。なお，ここで HiRDB を開始しようとしてもサービスが登録されていないため，開始できません。
3. アンインストール※してください。ただし，このときは DLL や EXE ファイルは残ったままの状態となります。
4. 残っているファイルを削除します。後でインストールしたときに回復させるため，定義情報がある CONF ディレクトリ下にあるファイルを退避しておいてください。その後，インストールしたディレクトリを削除してください（省略値でインストールしている場合は，HiRDB/ シングルサーバのときは C:¥win32app¥hitachi¥hirdb_s，HiRDB/ パラレルサーバのときは C:¥win32app¥hitachi¥hirdb_p となります）。
5. Windows を再起動してください。
6. 通常と同じ要領でインストールして Windows を再起動してください。
7. 退避していた HiRDB システム定義ファイルを %PDDIR%¥conf ディレクトリ下にコピーしてください。
8. HiRDB 開始準備（システムファイルなどの割り当て，初期化）をしてから開始してください。

1. ～ 5. を実行すると，異常状態のレジストリ及びサービス登録を正常状態（初期状態）に戻します。6. 以降は通常のインストール手順となります。

注※

アンインストール時にクライアント用の定義情報 hirdb.ini ファイルも削除されるため，必要に応じて退避しておいてください。

# 2.3 インストール後の作業

ここでは，製品をインストールした後の作業について説明します。

## 2.3.1 文字コードの選択

HiRDB で使用する文字コードを選択します。新規インストール後はシフト JIS 漢字コードが設定されています。単一バイト文字コード，Unicode（UTF-8），EUC 中国語漢字コード，及び中国語漢字コード（GB18030）を使用する場合は，pdntenv コマンドで使用する文字コードを変更してください。

なお，HiRDB Text Search Plug-in Version 2 02-02 以前のバージョンでは，Unicode（UTF-8）を使用できません。そのため，02-02 以前の HiRDB Text Search Plug-in Version 2 と組み合わせて使用する場合は，UTF-8 を指定しないでください。

**注意事項**

データベースの構築後に文字コードを変更する場合は，データベースを再構築する必要があります。

## 2.3.2 HiRDB 運用ディレクトリ下のファイルの削除

サーバプロセス，又はクライアントの強制終了時などに，HiRDB は %PDDIR%¥spool 下にトラブルシュート情報を出力します。また，ワークファイルの出力先を特に指定していない場合にコマンド又はユティリティを［Ctrl ＋ C］キーを押すなどして途中終了させると，%PDDIR%¥tmp 下にコマンド又はユティリティが出力した作業用一時ファイルが削除されないで残ります。これらのファイルを残しておくと，HiRDB 運用ディレクトリがあるディスクの容量を圧迫する原因になります。HiRDB 運用ディレクトリがあるディスクの容量が不足すると HiRDB が異常終了することがあるため，HiRDB は次に示すファイルを定期的に削除します。

- トラブルシュート情報ファイル（%PDDIR%¥spool 下のファイル）
- 作業用一時ファイル（%PDDIR%¥tmp 下のファイル）
- pd_tmp_directory オペランドに指定したディレクトリ下のファイル

これらの単調増加するファイルについては，「1.2.2　単調増加ファイル」を参照してください。

なお，通常はこれらのファイルを 24 時間ごとに削除します。この削除間隔を pd_spool_cleanup_interval オペランドで変更できます。また，pd_spool_cleanup_interval_level オペランドで指定した日より前に出力されたファイルだけを削除するという指定ができます。

このほかにも，トラブルシュート情報（%PDDIR%¥spool 下のファイル）を一括して削除する方法があります。

- pdcspool コマンドでトラブルシュート情報ファイルを削除できます。作業用一時ファイル（%PDDIR%¥tmp 下のファイル）も削除できます。
- HiRDB の開始時に自動的にトラブルシュート情報ファイルを削除します。pd_spool_cleanup オペランドでトラブルシュート情報ファイルを削除するかどうかを指定します。このオペランドの省略値は「削除する」です。また，pd_spool_cleanup_level オペランドで指定した日より前に出力されたトラブルシュート情報ファイルだけを削除するという指定ができます。

**備考**

pdcspool コマンドのオプション，pd_spool_cleanup_level，又は pd_spool_cleanup_interval オペランドの指定で，削除するトラブルシュート情報を選択できます。

## 2.3.3 ワークファイル出力先ディレクトリの作成

**実行者** HiRDB 管理者

HiRDB が出力するワークファイルの出力先となるディレクトリを作成してください。コマンドやユティリティの実行時に生成される様々なワークファイルの出力先として，ここで作成したディレクトリを指定します。これによって，出力先がユニット単位で 1 か所になるため，煩雑になりやすいワークファイルの管理が容易になります。ワークファイル出力先ディレクトリを作成しない運用もできますが，その場合はワークファイルの出力先が一定にならないため，pdcspool コマンドによるワークファイルの削除ができなくなります。そのため，ワークファイル出力先ディレクトリを作成しておくことをお勧めします。

### (1) ワークファイル出力先ディレクトリの容量の見積もり

ワークファイル出力先ディレクトリ下の空き領域は，次の値以上に設定してください。ワークファイル出力中に HiRDB 又はコマンドが異常終了した場合，ワークファイルは削除されません。そのため，pdcspool コマンドを実行する前にディスク容量が不足しないように，ワークファイル出力先ディレクトリの空き領域には十分に余裕のある値を設定してください。

ワークファイル出力先ディレクトリの容量（単位：キロバイト）＝
178224 ＋ a ＋ b ＋ c ＋ e ＋ f ＋ g ＋ h

| 変数 | 説明 | 計算式，又は計算式の参照箇所 |
|---|---|---|
| a | pdconstck 実行時の処理結果ファイルの容量 | 「19.1.10　整合性チェックユティリティ（pdconstck）実行時のファイルの容量」 |
| b | pddbst 実行時の次のファイルの容量<br>• ワーク用ファイル<br>• ソート用ワークファイル | 「19.1.4　データベース状態解析ユティリティ（pddbst）実行時のファイルの容量」 |
| c | pdload 実行時の次のファイルの容量<br>• インデクス情報ファイル<br>• エラー情報ファイル<br>• エラー情報ファイル作成用一時ファイル<br>• LOB 中間ファイル<br>• ソート用ワークファイル | 「19.1.1　データベース作成ユティリティ（pdload）実行時のファイルの容量」 |
| e | pdrbal 実行時の次のファイルの容量<br>• インデクス情報ファイル<br>• ソート用ワークファイル | 「19.1.9　リバランスユティリティ（pdrbal）実行時のファイルの容量」 |
| f | pdrorg 実行時の次のファイルの容量<br>• インデクス情報ファイル<br>• ソート用ワークファイル | 「19.1.2　データベース再編成ユティリティ（pdrorg）実行時のファイルの容量」 |
| g | pdrstr -w 実行時のソート用ワークディレクトリの容量 | マニュアル「HiRDB Version 8 コマンドリファレンス」 |
| h | pdstedit 実行時の次のファイルの容量<br>• ワーク用ファイル<br>• ソート用ワークファイル<br>• DAT 形式ファイル | 「19.1.3　統計解析ユティリティ（pdstedit）実行時のファイルの容量」 |

### (2) ワークファイル出力先ディレクトリの指定

ワークファイルの出力先を 1 か所にするには，pd_tmp_directory オペランドに作成したディレクトリを指定します。

pd_tmp_directory オペランドを指定していない場合，HiRDB は各コマンド及びユティリティによって決められたディレクトリにワークファイルを出力します。なお，コマンドやユティリティ実行時のワークファイル出力先は次の順番で決定されます。

1. コマンドのオプション，又はユティリティの制御文で指定した出力先
2. 1. の指定がない場合，pd_tmp_directory オペランドで指定した出力先
3. 2. の指定がない場合，システム環境変数 TMP で指定した出力先
4. 3. の指定がない場合，%PDDIR%tmp

### (3) ワークファイルの削除

HiRDB は通常，24 時間ごとにワークファイルを削除します。この削除間隔を pd_spool_cleanup_interval オペランドで変更できます。また，pd_spool_cleanup_interval_level オペランドで指定した日より前に出力されたファイルだけを削除するという指定ができます。このとき，pd_tmp_directory オペランドで指定したワークファイル出力先ディレクトリ下のファイルを削除します。

また，コマンドやユティリティが出力したワークファイルのうち，HiRDB が削除しないものについては，pdcspool コマンドによって定期的に削除する必要があります。この場合にも，pd_tmp_directory オペランドで指定したワークファイル出力先ディレクトリ下のファイルを削除します。

ワークファイルの削除については，「2.3.2　HiRDB 運用ディレクトリ下のファイルの削除」を参照してください。

## 2.3.4　HiRDB ファイルシステム領域を作成する準備

HiRDB ファイルシステム領域を作成する前に必要な作業，HiRDB ファイルシステム領域を作成するときの手順，及び HiRDB ファイルシステム領域のアクセス権の考え方について説明します。

### (1) 事前準備

**実行者**　Administrator

次の作業を行います。

- ハードディスクを用意します。
- 初期化したハードディスクにパーティションを設定します。
- NTFS 領域上に HiRDB ファイルシステム領域を作成する場合は，ディスクの初期化を行います。

これらの作業方法については，OS のマニュアルを参照してください。

### (2) HiRDB ファイルシステム領域の作成

**実行者**　HiRDB 管理者

pdfmkfs コマンドを実行して，NTFS 領域上に HiRDB ファイルシステム領域を作成します。又は，ダイレクトディスクアクセスを使用する場合，ディスクパーティション（論理ドライブ）上に HiRDB ファイルシステム領域を作成します。

### (3) HiRDB ファイルシステム領域のアクセス権の考え方

HiRDB ファイルシステム領域は，OS 上のアクセス権が有効になりますが，HiRDB ファイルに対しては OS 上のアクセス権は無効です。また，HiRDB では，HiRDB ファイルのアクセス権の制御は行っていません。したがって，HiRDB ファイルごとにアクセスを制限したい場合は，HiRDB ファイルシステム領域を切り分けて，HiRDB ファイルシステム領域ごとにアクセス権を変更してください。

### 2.3.5 Windows ファイアウォールの設定を有効にしている場合

Windows ファイアウォールの設定を有効にしている場合，例外リストに HiRDB が使用するポート番号やプログラムを登録する必要があります。登録方法については，「2.6 Windows ファイアウォールの例外リストへの登録」を参照してください。

# 2.4 付加 PP のインストール時の注意

付加 PP をインストールする場合の，規則及び注意について説明します。

HiRDB の付加 PP の機能を使用する場合は，付加 PP をインストールします。付加 PP の機能及び付加 PP をインストールするサーバを次の表に示します。

**表 2-4 付加 PP の機能とインストール先サーバ**

| 製品名 | 付加 PP を導入すると使用できる機能 | インストール先サーバマシン |
|---|---|---|
| HiRDB Advanced High Availability | • 1：1 スタンバイレス型系切り替え機能<br>• 影響分散スタンバイレス型系切り替え機能 | すべてのサーバマシン |
| | グローバルバッファの動的変更（pdbufmod コマンド） | |
| | システム構成変更コマンド（pdchgconf コマンド） | |
| HiRDB Advanced Partitioning Option | 表のマトリクス分割 | |
| | 分割格納条件の変更（ALTER TABLE） | |
| HiRDB Non Recover FES | 回復不要 FES | |
| HiRDB Accelerator | インメモリデータ処理 | |
| HiRDB LDAP Option | Sun Java System Directory Server 連携機能 | ディクショナリサーバがあるサーバマシン |

## (1) 規則及び注意事項

1. 付加 PP はキーモジュール（HiRDB LDAP Option はライブラリ）で提供されます。
2. 付加 PP のキーモジュールは，HiRDB インストールディレクトリの直下にコピーされます。HiRDB LDAP Option のライブラリは，%PDDIR%¥hirdb_ldap_sods ディレクトリ下にコピーされます。
3. 付加 PP をインストールする場合は，HiRDB サービスを停止した状態で実行してください。HiRDB サービスが起動中の場合はインストールを中止します。
4. HiRDB をバージョンアップした場合は，HiRDB のバージョンに対応する付加 PP をインストールする必要があります。また，HiRDB のインストールディレクトリを変更した場合は，付加 PP を再インストールする必要があります。
5. 付加 PP のインストールを実行できるのは，Administrators グループのユーザです。
6. HiRDB がインストールされていないと，付加 PP はインストールできません。
7. 付加 PP をアンインストールする場合は，HiRDB サービスを停止した状態で実行してください。アンインストールすると，インストール時にコピー及び設定したすべての情報が削除されます。
8. セットアップ識別子付きの HiRDB に付加 PP をインストールする場合は，インストール先の HiRDB と同じセットアップ識別子を指定して付加 PP をインストールしてください。

# 2.5 HiRDB のアンインストール手順

HiRDB のアンインストールは，今後このサーバマシンで HiRDB を使用しないときだけ実施してください。それ以外でアンインストールを実施することはお勧めしません。

## (1) HiRDB をアンインストールするユニットの状態を確認

HiRDB をアンインストールする前に，HiRDB をアンインストールするユニットの状態を確認します。該当するユニットで pdls -d ust コマンドを実行します。

- 終了ステータスが 0 の場合（ユニットの状態が ONLINE）：
  HiRDB が稼働中です。HiRDB を pdstop コマンドで正常終了させてください。
- 終了ステータスが 4 の場合（ユニットの状態が STARTING，又は STOPPING）：
  HiRDB が開始処理の途中，又は停止処理の途中です。処理が終了してから pdls -d ust コマンドを再度実行してください。
- 終了ステータスが 8 の場合（ユニットの状態が PAUSE）：
  障害によって，プロセスサーバプロセスの再起動を中断した状態です。この状態で，アンインストールを実行する場合は，「(3)HiRDB のアンインストール」へ進んでください。
  アンインストールを実行しない場合は，KFPS00715-E メッセージ及びこのメッセージ以前のイベントログに出力されたメッセージを参照して障害の原因を取り除いてから，サービスを開始し，ユニットを再開始してください。
- 終了ステータスが 12 の場合の場合（ユニットの状態が STOP）：
  HiRDB が停止状態です。サービスを停止してください。
- 終了ステータスが 16 の場合（セットアップ状態が UNSETUP）：
  アンインストールを実行できます。

## (2) HiRDB のサービスの停止

アンインストールする HiRDB のサービスが開始しているかどうかを［コントロールパネル］－［サービス］で確認してください。開始している場合は HiRDB のサービスを停止してから，アンインストールを実行してください。

## (3) HiRDB のアンインストール

HiRDB をアンインストールする場合は，コマンド，ユティリティ，アプリケーション，HiRDB Datareplicator，及び HiRDB Dataextractor はあらかじめ停止しておいてください。これらを停止しないと，実行形式ファイルや共用ライブラリの削除に失敗することがあります。

**注意事項**

アンインストールを行う前に，Windows ファイアウォールの例外リストから HiRDB が使用するポート番号やプログラムを削除してください。アンインストールを行った後では正しく削除できません。
削除する前にアンインストールしてしまった場合は，HiRDB を再度インストールし，その後に例外リストから削除する必要があります。
例外リストからの削除方法については，「2.7 Windows ファイアウォールの例外リストからの削除」を参照してください。

### (a) HiRDB のアンインストール

HiRDB/ シングルサーバのアンインストール手順を次に示します。

**〈手順〉**

1. ［スタート］－［プログラム］－［HiRDBSingleServer］－［HiRDBSingleServer のアンインストール］を選択します。
   ［HiRDBSingleServer のアンインストール］が登録されていないバージョンの場合は，［コントロールパネル］－［アプリケーションの追加と削除］の［インストールと削除］を表示します。一覧の中から「HiRDB/SingleServer」を選択し，［追加と削除］をクリックし，［OK］ボタンをクリックします。
2. アンインストールが始まります。しばらくすると，［共有ファイルを削除しますか？］という問い合わせのメッセージが表示されることがあります。表示されたら，［いいえ］をクリックしてください。
3. アンインストールが終わると，「HiRDB/Single Server の再インストールは，コンピュータを再起動した後に行ってください」というメッセージが表示されます。［OK］ボタンをクリックしてください。
4. 次に［コンピュータからプログラムを削除］ダイアログボックスの［OK］ボタンをクリックしてください。これで，アンインストールは終了です。
   アンインストールが成功していない場合，考えられる原因と対策を次に示します。

| 考えられる原因 | 対策 |
|---|---|
| コマンド，ユティリティ，HiRDB Datareplicator，又は HiRDB Dataextractor のどれかが実行されています。 | コマンド，ユティリティ，HiRDB Datareplicator，又は HiRDB Dataextractor を停止して，アンインストールを再度実行してください。 |
| そのほか（障害が発生しているなど） | イベントログに出力されたメッセージを参照して，障害原因を取り除いてから，アンインストールを再度実行してください。 |

5. アンインストールしても，インストール後に作成されたディレクトリ及びファイルは，インストールディレクトリ下に残ります。インストールディレクトリがデフォルトの場合は，エクスプローラなどを使用して，「C:¥win32app¥hitachi」ディレクトリ下の「hirdb_s」ディレクトリを削除してください。ただし，HiRDB Control Manager - Agent をアンインストールしていない場合には，先に HiRDB Control Manager - Agent をアンインストールしてから，「hirdb_s」ディレクトリを削除してください。

### (b) HiRDB/ パラレルサーバのアンインストール

HiRDB/ パラレルサーバをアンインストールする場合は，HiRDB/ パラレルサーバを構成するすべてのサーバマシンでアンインストールを実施します。アンインストール手順を次に示します。

**〈手順〉**

1. ［スタート］－［プログラム］－［HiRDBParallelServer］－［HiRDBParallelServer のアンインストール］を選択します。
   ［HiRDBParallelServer のアンインストール］が登録されていないバージョンの場合は，［コントロールパネル］－［アプリケーションの追加と削除］の［インストールと削除］を表示します。一覧の中から「HiRDB/ParallelServer」を選択し，［追加と削除］をクリックし，［OK］ボタンをクリックします。
2. アンインストールが始まります。しばらくすると，［共有ファイルを削除しますか？］という問い合わせのメッセージが表示されることがあります。表示されたら，［いいえ］をクリックしてください。
3. アンインストールが終わると，「HiRDB/Parallel Server の再インストールは，コンピュータを再起動した後に行ってください」というメッセージが表示されます。［OK］ボタンをクリックしてください。

4. 次に［コンピュータからプログラムを削除］ダイアログボックスの［OK］ボタンをクリックしてください。これで，アンインストールは終了です。
   アンインストールが成功していない場合，考えられる原因と対策を次に示します。

| 考えられる原因 | 対策 |
|---|---|
| コマンド，ユティリティ，HiRDB Datareplicator，又は HiRDB Dataextractor のどれかが実行されています。 | コマンド，ユティリティ，HiRDB Datareplicator，又は HiRDB Dataextractor を停止して，アンインストールを再度実行してください。 |
| そのほか（障害が発生しているなど） | イベントログに出力されたメッセージを参照して，障害原因を取り除いてから，アンインストールを再度実行してください。 |

5. アンインストールしても，インストール後に作成されたディレクトリ及びファイルは，インストールディレクトリ下に残ります。インストールディレクトリがデフォルトの場合は，エクスプローラなどを使用して，「C:¥win32app¥hitachi」ディレクトリ下の「hirdb_p」ディレクトリを削除してください。ただし，HiRDB Control Manager - Agent をアンインストールしていない場合には，先に HiRDB Control Manager - Agent をアンインストールしてから，「hirdb_p」ディレクトリを削除してください。

### (4) 注意事項

HiRDB をアンインストールした後に，HiRDB 運用ディレクトリ下にファイルが残ることがあります。この場合，ファイルを削除してください。

# 2.6 Windows ファイアウォールの例外リストへの登録

Windows ファイアウォールを有効にしている環境で HiRDB を使用する場合，HiRDB が使用するポート番号やプログラムを例外リストに登録してください。ここでは，その登録方法について説明します。

次に示すプログラム及びポート番号を例外リストに登録します。

- HiRDB サーバのプログラム
- リモートシェル実行時に使用するポート番号
- UAP

**注意事項**

Windows ファイアウォールを有効にしている環境で，前記のプログラムやポート番号を例外リストに登録しないと，HiRDB が開始できない，UAP が HiRDB に接続できないなどの不具合が起きることがあります。

**参考**

HiRDB/ シングルサーバで，HiRDB サーバと HiRDB クライアントが同一マシン上にある場合，設定によって例外リストへの登録が不要になります。詳細については，「2.6.4　例外リストへの登録が不要になるケース」を参照してください。

## 2.6.1 HiRDB サーバのプログラムを例外リストに登録する

HiRDB サーバのプログラムを例外リストに登録します。

### (1) 前提条件

HiRDB のインストール時に，［Windows ファイアウォールへの登録］画面で HiRDB サーバのプログラムを Windows ファイアウォールの例外リストに登録済みの場合は，(2) で説明する作業は必要ありません。「2.6.2　リモートシェル実行時に使用するポート番号を例外リストに登録する」にお進みください。

### (2) 登録方法

HiRDB コマンドプロンプトを起動し，次に示すバッチファイルを実行してください。

```
%PDDIR%¥BIN¥pdsetfw.bat "セットアップ識別子"
```

インストール時に指定したセットアップ識別子を指定してください。標準インストールを行った場合は，セットアップ識別子を省略してください。

このバッチファイルを実行すると，Windows ファイアウォールの例外リストに HiRDB サーバのプログラムが登録されます。

**参考**

Windows ファイアウォールの［例外］タブを参照すると，HiRDB サーバのプログラムが登録されていることが確認できます。

## 2.6.2 リモートシェル実行時に使用するポート番号を例外リストに登録する

リモートシェル実行時に使用するポート番号を例外リストに登録します。

### (1) 前提条件

次に示す条件を一つでも満たす場合は，(2) で説明する作業を行ってください。該当しない場合は，ここで説明する作業は必要ありません。「2.6.3　UAP を例外リストに登録する」にお進みください。

- HiRDB/ パラレルサーバの場合
- IP アドレスを引き継がない系切り替え機能を使用する場合

### (2) 登録方法

登録手順を次に示します。

**手順**

1. コントロールパネルから Windows ファイアウォールを起動します。
2. ［例外］タブの［ファイルとプリンタの共有］を選択し，［編集］ボタンをクリックします。
3. ［TCP 445］のチェックボックスをチェックし，［スコープの変更］ボタンをクリックします。
4. ［スコープの変更］で，［ユーザーのネットワーク（サブネット）のみ］又は［カスタムの一覧］で関連する PC を設定します。

さらに，ドメインに属していないサーバマシンで HiRDB/ パラレルサーバを構成している場合は，HiRDB 運用ディレクトリを次に示す手順で共有化します。

**手順**

1. エクスプローラで HiRDB 運用ディレクトリを右クリックし，［共有とセキュリティ］を選択します。
2. ［ネットワーク上での共有とセキュリティ］の［ネットワーク上でこのフォルダを共有する］及び［ネットワークユーザによるファイルの変更を許可する］のチェックボックスをチェックします。

## 2.6.3 UAP を例外リストに登録する

UAP を例外リストに登録する方法は二つあります。どちらかの方法で例外リストに登録してください。

**● ポート番号を例外リストに登録する方法**

クライアント環境定義の PDCLTRCVPORT オペランドに指定したポート番号（HiRDB サーバと HiRDB クライアントが通信を行う場合に使用する受信ポート番号）をすべて例外リストに登録します。

**● UAP の実行ファイルを例外リストに登録する方法**

コントロールパネルから Windows ファイアウォールを起動して，［例外］タブを選択し，［プログラムの追加］から UAP の実行ファイルをすべて登録してください。

### 2.6.4 例外リストへの登録が不要になるケース

HiRDB/ シングルサーバで，HiRDB サーバと HiRDB クライアントが同一マシン上にある場合，次に示す設定を行うと Windows ファイアウォールの例外リストへの登録が不要になります。

- pdunit オペランドの -x オプションにループバックアドレス（127.0.0.1）を指定する
- pd_rpc_bind_loopback_address オペランドに Y を指定する
- クライアント環境定義の PDHOST オペランドにループバックアドレスを指定する
- クライアント環境定義の PDCLTBINDLOOPBACKADDR オペランドに YES を指定する

**簡易セットアップツール使用時の注意事項**

簡易セットアップツールを使用して HiRDB/ シングルサーバの環境設定をする場合は，次のように作業してください。

- HiRDB/ シングルサーバを構築するマシンで簡易セットアップツールを実行する
- 簡易セットアップツールの開始画面でホスト名にループバックアドレスを指定する
- 簡易セットアップツールの詳細定義画面で pd_rpc_bind_loopback_address オペランドに Y を指定する

このように作業しない場合，HiRDB サーバのプログラムが Windows ファイアウォールによってブロックされるため，簡易セットアップツールを実行しても HiRDB を開始できません。

# 2.7 Windows ファイアウォールの例外リストからの削除

HiRDB をアンインストールする場合，Windows ファイアウォールの例外リストに登録した HiRDB が使用するポート番号やプログラムを削除する必要があります。ここでは，その削除方法について説明します。

**注意事項**

アンインストールを行う前に，HiRDB が使用するポート番号やプログラムを例外リストから削除してください。アンインストールを行った後では正しく削除できません。削除する前にアンインストールしてしまった場合は，HiRDB を再度インストールし，その後に例外リストから削除してください。

## 2.7.1 HiRDB サーバのプログラムを例外リストから削除する

HiRDB サーバのプログラムを例外リストから削除します。

### (1) 前提条件

HiRDB サーバのプログラムを「2.6.1　HiRDB サーバのプログラムを例外リストに登録する」の説明に従って例外リストに登録した場合に，(2) で説明する作業を行ってください。

HiRDB のインストール時に，［Windows ファイアウォールへの登録］画面で HiRDB サーバのプログラムを例外リストに登録した場合は，(2) で説明する作業は必要ありません。「2.7.2　リモートシェル実行時に使用するポート番号を例外リストから削除する」にお進みください。

**参考**

インストール時に HiRDB サーバのプログラムを例外リストに登録した場合は，HiRDB をアンインストールするときに，例外リストから HiRDB サーバのプログラムが自動的に削除されます。

### (2) 削除方法

HiRDB コマンドプロンプトを起動し，次に示すバッチファイルを実行してください。

```
%PDDIR%¥BIN¥pddelfw.bat
```

このバッチファイルを実行すると，Windows ファイアウォールの例外リストから HiRDB サーバのプログラムが削除されます。

## 2.7.2 リモートシェル実行時に使用するポート番号を例外リストから削除する

リモートシェル実行時に使用するポート番号を例外リストから削除します。

### (1) 前提条件

リモートシェル実行時に使用するポート番号を「2.6.2　リモートシェル実行時に使用するポート番号を例

外リストに登録する」の説明に従って例外リストに登録した場合に，(2) で説明する作業を行ってください。該当しない場合は，ここで説明する作業は必要ありません。「2.7.3　UAP を例外リストから削除する」にお進みください。

### (2) 削除方法

削除手順を次に示します。

**手順**

1. コントロールパネルから Windows ファイアウォールを起動します。
2. ［例外］タブの［ファイルとプリンタの共有］を選択し，［編集］ボタンをクリックします。
3. ［TCP 445］のチェックボックスをオフにします。

**参考**

この設定を行うと HiRDB 以外のプログラムで支障が発生する場合は，この設定を行う必要はありません。

## 2.7.3　UAP を例外リストから削除する

UAP を例外リストから削除します。

### (1) 前提条件

UAP を「2.6.3　UAP を例外リストに登録する」の説明に従って例外リストに登録した場合に，(2) で説明する作業を行ってください。該当しない場合は，ここで説明する作業は必要ありません。

### (2) 削除方法

削除方法を次に示します。

**● ポート番号を例外リストに登録した場合**

クライアント環境定義の PDCLTRCVPORT オペランドに指定したポート番号（HiRDB サーバと HiRDB クライアントが通信を行う場合に使用する受信ポート番号）をすべて例外リストから削除します。

**● UAP の実行ファイルを例外リストに登録した場合**

コントロールパネルから Windows ファイアウォールを起動して，［例外］タブを選択します。［プログラムおよびサービス］に登録した UAP の実行ファイルを選択し，削除してください。

**参考**

HiRDB 以外のプログラムがこれらのポート番号又は UAP を使用する場合は，例外リストから削除する必要はありません。

# 3 簡易セットアップツールによる環境設定

この章では，簡易セットアップツールを使用した HiRDB サーバの環境設定方法について説明します。


3.1 簡易セットアップツールの概要

3.2 簡易セットアップツールの起動

3.3 標準セットアップ

3.4 ウィザードセットアップ

3.5 カスタムセットアップ

3.6 カスタムセットアップ（詳細定義）

3.7 詳細定義情報の読み込み

3.8 定義の更新

3.9 HiRDB システム定義を編集する場合の注意事項

3.10 HiRDB/ パラレルサーバの環境設定をする場合

3.11 系切り替え機能の設定

# 3.1 簡易セットアップツールの概要

ここでは，簡易セットアップツールを使ってできることと，簡易セットアップツールの稼働環境について説明します。

## 3.1.1 簡易セットアップツールとは

簡易セットアップツールとは，HiRDB の環境構築を支援するツールで，GUI で次に示す作業を実施できます。

**● 環境設定**

簡易セットアップツールでは，次のどれかを選択して環境設定ができます。

- 標準セットアップ
  環境設定をするマシンのホスト名，HiRDB 運用ディレクトリ，及びセットアップディレクトリだけを指定すれば容易に環境設定ができます。
- ウィザードセットアップ
  システム構成，ユニット構成，及びシステムログの運用方法を，対話形式で設定できます。
- カスタムセットアップ
  HiRDB の規模，HiRDB 識別子，ポート番号，ユニット識別子，システムファイル及び RD エリアの作成ディレクトリなど，より詳細な設定ができます。

**● 定義の更新**

作成済みの %PDDIR%¥conf 下のシステム定義を更新します。

**● 定義情報の保存と読み込み**

簡易セットアップツールで作成した詳細定義情報を単一ファイル形式，又はディレクトリ形式で保存できます。また，保存した詳細定義情報を読み込むことができます。

**注意事項**

簡易セットアップツールで環境設定をすると，デフォルトでは RD エリアの自動増分機能，及びシステムログファイルの自動拡張機能が適用されます。RD エリアの自動増分，及びシステムログファイルの自動拡張については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。
なお，これらの機能を使用しない場合は，カスタムセットアップで適用を解除してください。解除する方法を次に示します。

≪ RD エリアの自動増分機能の適用を解除する場合≫
［HiRDB ファイルシステム領域編集］ダイアログボックスで「領域を自動増分する」のチェックを外してください。

≪システムログファイルの自動拡張機能の適用を解除する場合≫
［HiRDB セットアップツール－詳細定義（セットアップ）］ウィンドウで各サーバ定義を表示し，次のどちらかの設定をしてください。
・pd_log_auto_expand_size オペランドを削除する
・pd_log_auto_expand_size オペランドの 1 回当たりに拡張するサイズに 0 を指定する

## 3.1.2 環境設定

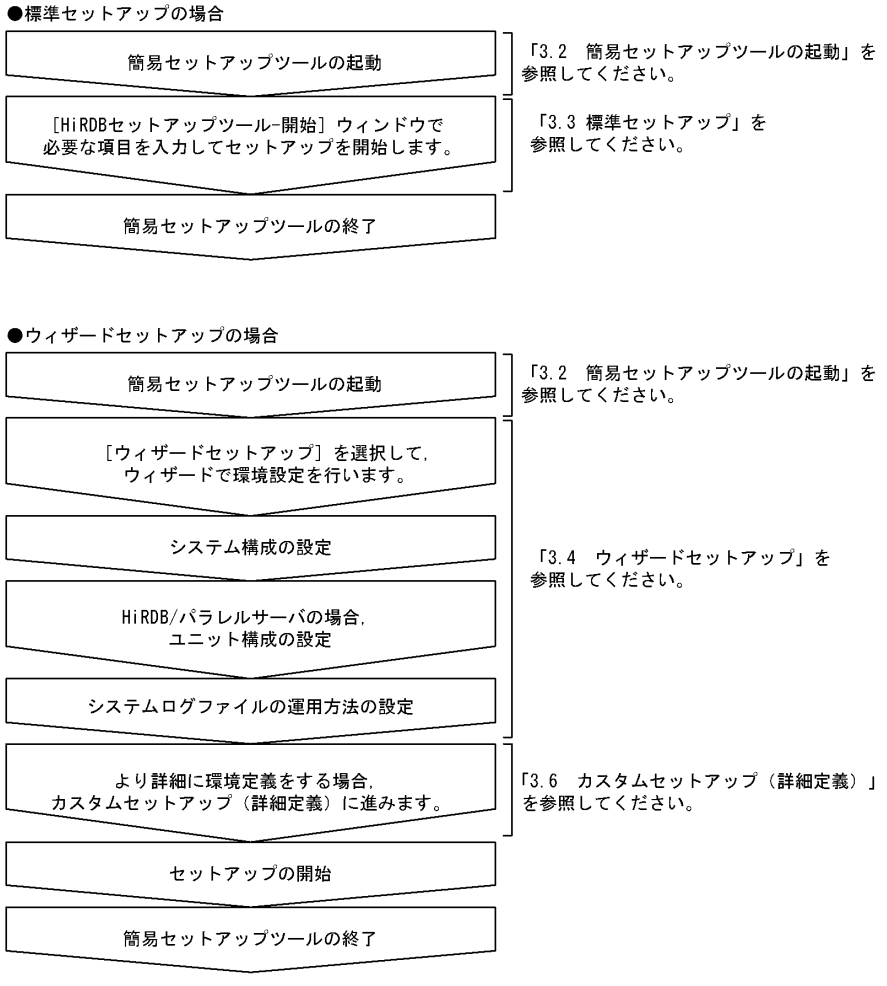
簡易セットアップツールで環境設定をすると，次のことができます。

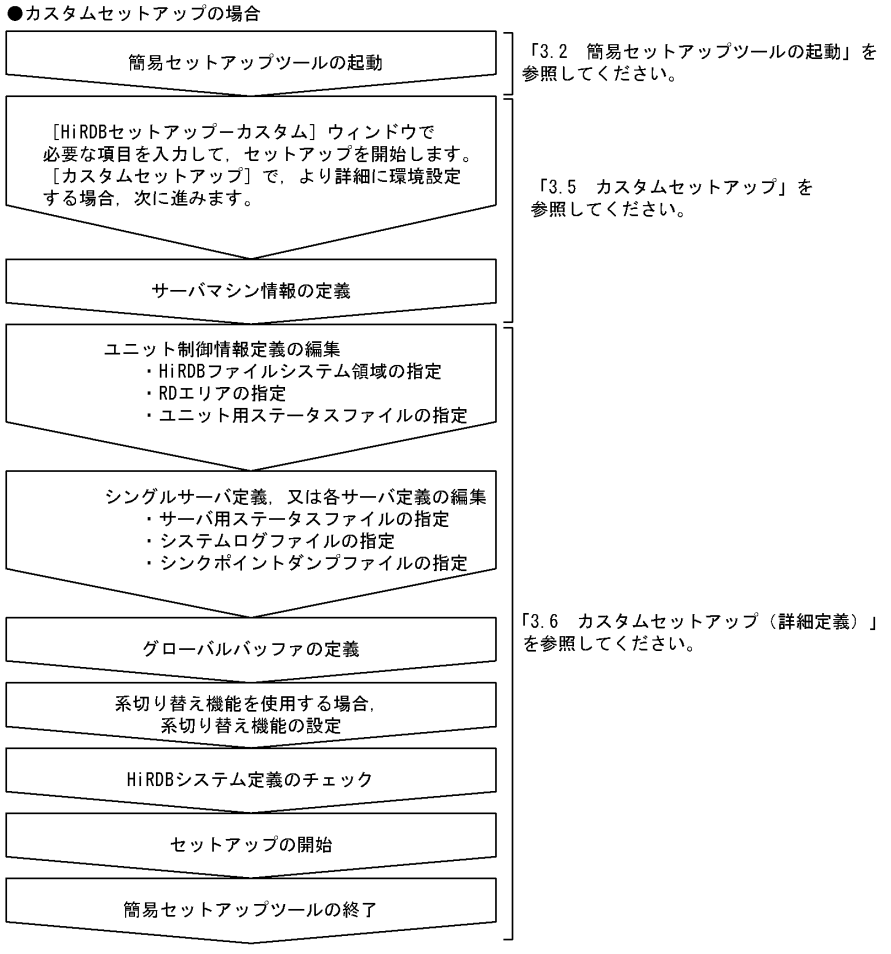
- HiRDB システム定義の作成
- HiRDB ファイルシステム領域の作成

- システムファイルの作成
- RD エリアの作成
- HiRDB の開始
- グローバルバッファの定義
- サンプルデータベースの作成

簡易セットアップツールによる環境設定手順を次の図に示します。

**図 3-1　簡易セットアップツールによる環境設定手順**






簡易セットアップツールを使用して環境設定をした場合に作成されるファイルを次に示します。

- システム定義ファイル
- バッチファイル及び制御文
- サンプルデータベース関連ファイル
- ログファイル
- バックアップファイル

それぞれのファイルについて説明します。

## (1) システム定義ファイル

%PDCONFPATH% 下に作成されます。作成されるシステム定義ファイル名を次に示します。

| 定義 | ファイル名 |
|---|---|
| システム共通定義 | pdsys |
| ユニット定義 | pdutsys |
| サーバ共通定義 | pdsvrc |

| 定義 | ファイル名 |
|---|---|
| シングルサーバ定義※1 | sds01※3 |
| ディクショナリサーバ定義※2 | dic1※3 |
| フロントエンドサーバ定義※2 | fes1※3 |
| バックエンドサーバ定義※2 | bes1，bes2※3 |

注※1
HiRDB/ シングルサーバの場合に作成されます。

注※2
HiRDB/ パラレルサーバの場合に作成されます。

注※3
サーバ名を変更することでファイル名は変更できます。サーバ名の変更については，「3.6.2(3)(b) サーバ名の設定」を参照してください。また，フロントエンドサーバ定義及びバックエンドサーバ定義はユーザが指定したサーバと同数のファイルを作成します。

**! 注意事項**

HiRDB システム定義ファイルのパーミッションは，ファイルの所有者（HiRDB 管理者）にだけ，読み込み権限及び書き込み権限を持たせるように設定，維持するようにしてください。

### (2) バッチファイル及び制御文

%PDCONFPATH% 下に作成されます。作成されるバッチファイル及び制御文のファイル名を次に示します。

| ファイル名 | 種別 | 使用するコマンド，又はユティリティ | 概要 |
|---|---|---|---|
| fmkfile.bat | バッチファイル | pdfmkfs コマンド | HiRDB ファイルシステム領域の初期設定 |
| fmkfs.bat | バッチファイル | pdfmkfs コマンド | HiRDB ファイルシステム領域の初期設定 |
| mkinit | 制御文 | create rdarea 文 | RD エリアの初期設定 |
| initdb.bat | バッチファイル | データベース初期設定ユティリティ（pdinit） | RD エリアの初期設定 |
| sysfint.bat | バッチファイル | pdstsinit コマンド<br>pdloginit コマンド | ステータスファイル，ログ関係ファイルの初期設定 |

### (3) サンプルデータベース関連ファイル

HiRDB のサンプルデータベースを作成するためのバッチファイル及びデータファイルは，環境設定の処理の中で自動的に %PDDIR%¥pdistup¥sample 下に作成されます。作成されるファイルを次に示します。

| ファイル名 | 種別 | 使用するユティリティ，又は SQL 文 | 概要 |
|---|---|---|---|
| PDI_SampleDef.bat | バッチファイル | データベース定義ユティリティ（pddef） | スキーマの定義 |
| PDI_SampleLod.bat | バッチファイル | データベース作成ユティリティ（pdload） | データのロード |
| PDI_CreateUser | DDL | 定義系 SQL の GRANT 文 | ユーザの定義 |
| PDI_CreateTable | DDL | 定義系 SQL の CREATE TABLE 文 | 表の定義 |
| PDI_Custom.csv | CSV | なし | 表 CUSTOM 用のデータ |
| PDI_Goods.csv | CSV | なし | 表 GOODS 用のデータ |
| PDI_Stock.csv | CSV | なし | 表 STOCK 用のデータ |
| PDI_Vendor.csv | CSV | なし | 表 VENDOR 用のデータ |

簡易セットアップツールで作成するサンプルデータベースについては，「3.3.3　簡易セットアップツールで作成するサンプルデータベース」を参照してください。

### (4) ログファイル

簡易セットアップツールは，HiRDB 起動処理中のエラーを少なくするために起動処理（pdstart コマンドの実行）前に定義をチェック（pdconfchk コマンドの実行）します。チェック結果がエラーとなった場合は，セットアップエラー（HiRDB の開始エラー）となり，チェック結果をログファイルに出力します。また，ログファイルには簡易セットアップツールの実行履歴も出力します。

ログファイル名は，%PDDIR%¥pdistup¥pdi_log.txt です。

セットアップ処理実行でエラーが発生した場合にはエラー情報がログファイルに出力されますが，出力されるログは中断の原因となった最終的なエラーです。イベントビューアにもエラー情報が出力されている可能性があるため，ログ情報とともにイベントビューアの内容を参照することをお勧めします。

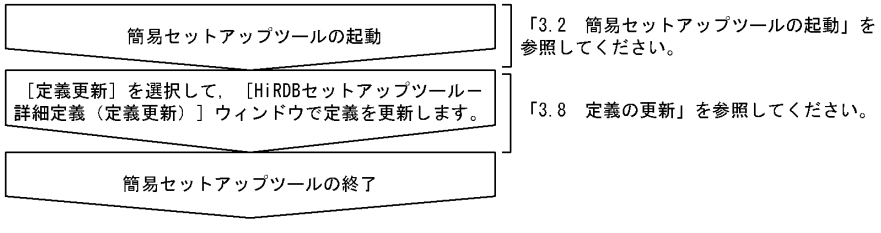
### (5) バックアップファイル

バックアップファイルを作成するディレクトリに同じ名称のファイルがある場合，既存ファイルのバックアップを作成します。バックアップファイルは，ファイル名に「_001 ～ 999」を付加します。ただし，_001 ～ 999 のすべてが既に使用されている場合はエラーになります。

## 3.1.3 定義の更新

簡易セットアップツールによる HiRDB システム定義の更新手順を次の図に示します。

**図 3-2　簡易セットアップツールによる HiRDB システム定義の更新手順**



## 3.1.4　簡易セットアップツールの稼働環境

Windows 版 HiRDB の環境設定をする場合は，HiRDB をインストールした Windows マシンで簡易セットアップツールを実行します。

**注意事項**

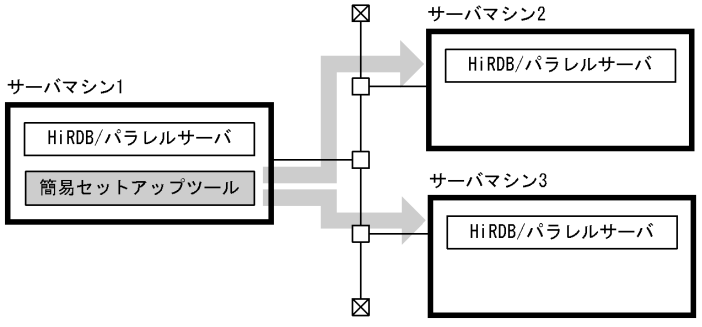
- 古いバージョンの簡易セットアップツールを使用しないでください。操作対象の HiRDB と同じバージョンの簡易セットアップツールを使用してください。
- マルチ HiRDB 環境を構築する場合，二つの HiRDB を簡易セットアップツールで同時に環境設定できません。一つの HiRDB を簡易セットアップツールで環境設定した後に，もう一つの HiRDB を簡易セットアップツールで環境設定してください。

### （1）Windows 版 HiRDB に対して簡易セットアップツールを使用する場合

HiRDB/ シングルサーバの環境設定を行う場合は，HiRDB/ シングルサーバをインストールしたサーバマシンで簡易セットアップツールを実行してください。HiRDB/ パラレルサーバの環境設定を行う場合は，HiRDB/ パラレルサーバを構成するサーバマシンの一つで簡易セットアップツールを実行してください。

簡易セットアップツールを使用した Windows 版 HiRDB の環境設定の概要を図 3-3 ～図 3-4 に示します。

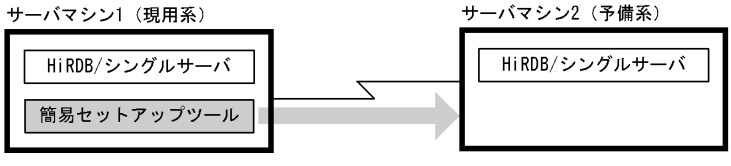
**図 3-3　簡易セットアップツールを使用した Windows 版 HiRDB の環境設定の概要（HiRDB/ パラレルサーバの場合）**




〔説明〕

サーバマシン 1 で簡易セットアップツールを実行し，サーバマシン 1，サーバマシン 2，及びサーバマシン 3 の HiRDB/ パラレルサーバの環境設定を行います。

**図 3-4　簡易セットアップツールを使用した Windows 版 HiRDB の環境設定の概要（HiRDB/ シングルサーバで系切り替え機能を使用する場合）**




〔説明〕

サーバマシン 1 で簡易セットアップツールを実行し，サーバマシン 1 を現用系に，サーバマシン 2 を

予備系に指定して HiRDB/ シングルサーバの環境設定を行います。

### (2) 簡易セットアップツールを実行するマシン環境

簡易セットアップツールを実行するマシンの前提 OS と，Windows 版 HiRDB のセットアップ可否を次の表に示します。

**表 3-1 簡易セットアップツールを実行するマシンの前提 OS と，Windows 版 HiRDB のセットアップ可否**

| プラットフォーム | 前提 OS | Windows 版 HiRDB のセットアップ |
|---|---|---|
| Windows | Windows 2000 | ○ |
| | Windows XP | ○ |
| | Windows Server 2003 | ○ |
| | Windows Vista Business | ○ |
| | Windows Vista Ultimate，又は Windows Vista Enterprise | ○ |
| | Windows Server 2008 | ○ |
| | Windows 7 Professional | ○ |
| | Windows 7 Ultimate，又は Windows 7 Enterprise | ○ |
| Windows(IPF) | Windows Server 2003 | ○ |
| | Windows Server 2008 | ○ |
| Windows (x64)<br>64 ビットモード※ | Windows XP | ○ |
| | Windows Server 2003 | ○ |
| | Windows Vista Business | ○ |
| | Windows Vista Ultimate，又は Windows Vista Enterprise | ○ |
| | Windows Server 2008 | ○ |
| | Windows 7 Professional | ○ |
| | Windows 7 Ultimate，又は Windows 7 Enterprise | ○ |

（凡例）
○：セットアップできます。

注※
32 ビットエミュレーションモードで動作します。

## 3.1.5 簡易セットアップツールを実行する前に確認すること

簡易セットアップツールを実行する前に次に示すことを確認してください。

### (1) Windows 版 HiRDB に対して簡易セットアップツールを使用する場合

次に示すことを確認してください。

- 操作対象の HiRDB のサービスが起動中であるかを確認してください。起動中でない場合はサービスを起動してください。
- 簡易セットアップツールを実行するサーバマシンと簡易セットアップツールを適用するサーバマシンの，HiRDB サービスのユーザ名とパスワードが一致しているか確認してください。

- 簡易セットアップツールを使用して HiRDB/ シングルサーバの環境を構築する場合，pdntenv コマンド（-ro オプションに on を指定）でリモート系コマンドを使用する設定を行ってください。その後，HiRDB サービスを再起動してください。
- 簡易セットアップツールを使用して環境構築を行うサーバマシンの SERVICES ファイルに，サービス名 pdrshsrv とポート番号 49999 を設定してください。セットアップ識別子付きでインストールした場合は，サービス名を pdrshsrv[ 識別子 ] としてください。ポート番号は，簡易セットアップツール以外のアプリケーションと重複しないようにしてください。SERVICES ファイルの設定例については，「2.1.3　OS 環境ファイルの設定」を参照してください。

### （2）HiRDB/ パラレルサーバに対して簡易セットアップツールを使用する場合

次に示すことを確認してください。

- ネットワーク形態は，単一ドメインの環境をお勧めします。
- 正常開始以外の開始モードで開始したサーバが一つ以上ある場合，HiRDB を一度正常終了してください。

### （3）Windows の設定について

Windows のフォントサイズが規定のサイズ（96dpi）であることを確認してください。

# 3.2 簡易セットアップツールの起動

## (1) 簡易セットアップツールを実行するユーザ

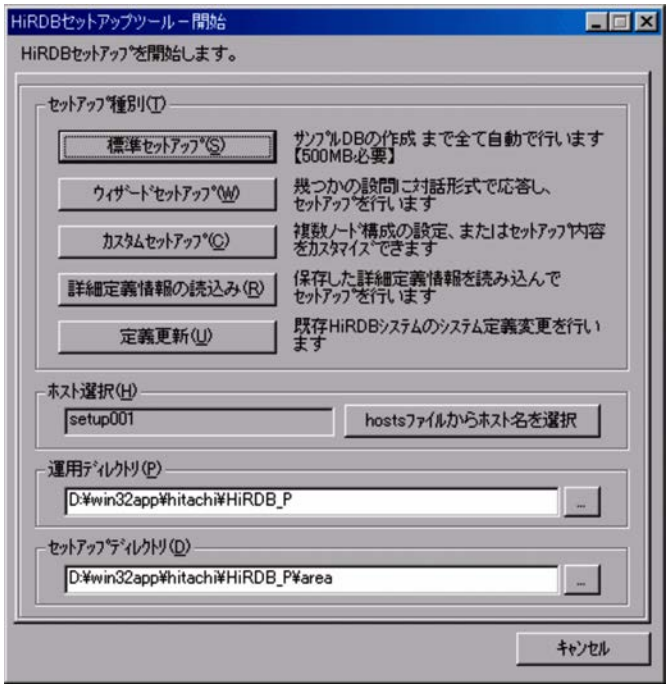
簡易セットアップツールは Administrators 権限を持っているユーザが実行できます。ただし，環境設定をする HiRDB のホスト名を指定するときに OS に登録している HiRDB 管理者用のユーザ名及びパスワードを指定してください。HiRDB 管理者のデフォルトは，HiRDB をインストールしたユーザになります。

## (2) 簡易セットアップツールの起動手順

簡易セットアップツールを起動方法には次の二つがあります。

1. HiRDB/ シングルサーバの場合，［スタート］－［プログラム］－［HiRDBSingleServer］－［簡易セットアップツール］を選択してください。
   HiRDB/ パラレルサーバの場合，［スタート］－［プログラム］－［HiRDBParallelServer］－［簡易セットアップツール］を選択してください。
2. %PDDIR%¥pdistup¥bin¥pdistup.exe を実行してください。

簡易セットアップツールを起動すると，［HiRDB セットアップツール－開始］ウィンドウが表示されます。




この章では，HiRDB/ パラレルサーバの場合のウィンドウで説明します。

**注意事項**

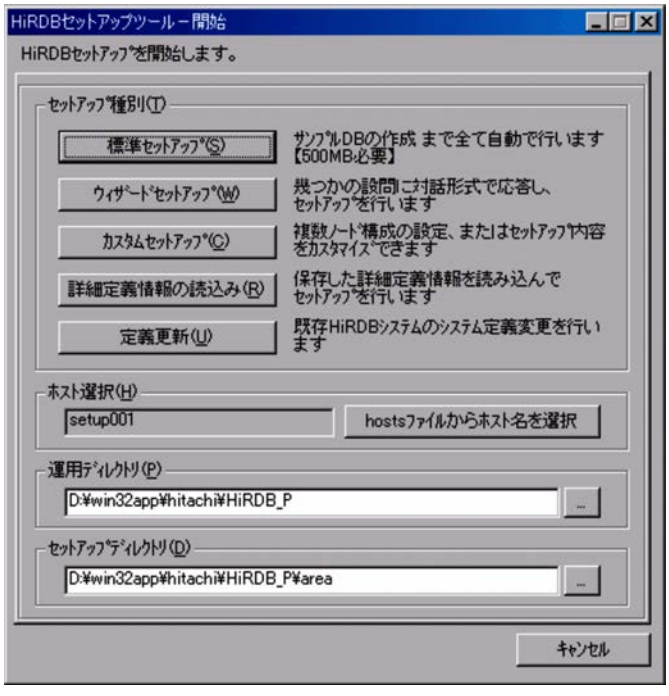
- 簡易セットアップツールは複数起動できません。
- 異なるマシンでそれぞれ簡易セットアップツールを起動し，同じホストに対して環境設定を実行しないでください。
- Windows Terminal Service を使用してセットアップツールを実行する場合，複数の Windows Terminal Service のクライアントから同時にセットアップツールを実行しないでください。
- 簡易セットアップツールが標準ホスト名の取得に失敗した場合，［ローカルマシン名入力］画面が表示されます。この場合，自マシン（簡易セットアップツールを実行するマシン）のホスト名を指定して，［OK］ボタンをクリックしてください。簡易セットアップツールの起動処理が再度行われます。

# 3.3 標準セットアップ

簡易セットアップツールの初期値で環境設定する場合は，［HiRDB セットアップツール一開始］ウィンドウから直接開始できます。環境設定をするマシンのホスト名，運用ディレクトリ，及びセットアップディレクトリの指定だけで，容易に HiRDB 環境を構築できます。

## 3.3.1 標準セットアップの場合の環境設定手順

［HiRDB セットアップツール一開始］ウィンドウで必要な項目を入力します。




標準セットアップの場合は，次に示す値で環境設定されます。

- HiRDB の規模：小規模
- システム ID：HRD1
- ポート番号：22200
- ユニット ID：UNT1
- サンプルデータベースの作成：作成する

標準セットアップの環境設定手順を次に示します。

1. 環境設定をするマシンのホスト名を指定します。
   ［ホスト選択］ラベルには，［hosts ファイルからホスト名を選択］ダイアログボックスで指定したホスト名がホスト名，FQDN，又は IP アドレスの形式で表示されます。
   初期値は HiRDB インストールディレクトリです。環境設定をするマシンのホスト名を変更する場合，［hosts ファイルからホスト名を選択］をクリックして，［hosts ファイルからホスト名を選択］ダイアログボックスで，ホスト名，Windows ディレクトリ，ユーザ名，及びパスワードを指定します。

それぞれの項目について説明します。

| 項目名 | 説明 |
|---|---|
| **ホスト名** | 環境設定をするマシンのホスト名を指定します。［hosts ファイルのホスト名一覧］リストから選択するか，又は直接入力してください。入力できる形式は，ホスト名，FQDN，又は IP アドレスです。また，簡易セットアップツールを実行しているマシンの場合は，ループバックアドレスも指定できます。<br>なお，ホスト名を変更した場合，［ユーザ名］及び［パスワード］の内容はリセットされるため，入力し直してください。 |
| Windows **ディレクトリ** | 環境設定をするマシンが Windows の場合は Windows インストールディレクトリをフルパスで指定します。通常は変更する必要はありません。セットアップ先が UNIX の場合は，指定は無視されます。<br>セパレータは ¥ と / のどちらも使用できますが，混在して指定できません。また，「**ドライブ名**:¥」と「/ **ドライブ名**」のどちらも指定できます。 |
| **ユーザ名** | 環境設定をするサーバマシンの HiRDB 管理者用のユーザ名を指定します。ドメインユーザの場合はドメインを付けて指定する必要はありません。 |
| **パスワード** | ［ユーザ名］に指定したユーザのパスワードを指定します。 |

なお，簡易セットアップツールで環境設定をすると，HiRDB に次に示す認可識別子を登録して DBA 権限を付与します。

- ［ホスト名］で選択したホストが UNIX の場合
  ユーザ：［ユーザ名］に指定した HiRDB 管理者用のユーザ名
  パスワード：［パスワード］に指定したパスワード
- ［ホスト名］で選択したホストが Windows の場合
  ユーザ：root
  パスワード：root

また，この設定のリモートホスト接続処理で，セットアップ先にワーク領域を作成します。ワーク領域には，セットアップ先に正常に接続しているかを確認するためのバッチファイルや，バッチファイルの実行結果を保存したファイルを格納します。ワーク領域は，次の場所に作成されます。

- ［ホスト名］で選択したホストが UNIX の場合
  /tmp/[ ログインユーザ名 ]
- ［ホスト名］で選択したホストが Windows の場合
  %windir%¥Temp¥[ ログインユーザ名 ]

2. ［運用ディレクトリ］テキストボックスで HiRDB 運用ディレクトリを指定します。セパレータは ¥ と / のどちらも使用できますが，混在して指定できません。また，「**ドライブ名** :¥」と「/ **ドライブ名**」のどちらも指定できます。初期値は HiRDB インストールディレクトリです※ 1。Windows の場合，HiRDB 運用ディレクトリとは，HiRDB のインストールディレクトリのことです。HiRDB 運用ディレクトリの指定を省略したり，存在しない HiRDB 運用ディレクトリを指定すると，エラーメッセージが表示されます。
［...］をクリックして，［フォルダの参照］ダイアログボックスからディレクトリを選択することもできます。
［フォルダの参照］ダイアログボックスに表示される内容は，環境設定をするホストの内容です。表示される項目を次に示します。
   - 未フォーマット状態のパーティション（ただし，ドライブ文字割り当て済み）
   - ドライブ
   - ディレクトリ
   - ファイル※ 2

   注※ 1
   %PDDIR%¥pdistup¥bin 下から簡易セットアップツールを起動しなかった場合は，空になります。

   注※ 2
   HiRDB ファイルシステム領域を編集する場合だけ表示されます。
3. ［セットアップディレクトリ］テキストボックスでセットアップディレクトリ（システムファイル及び RD エリアを作成するディレクトリ）を指定します。初期値は，%PDDIR%¥area です※。なお，Windows のショートパス名（例えば，C:¥PROGRA~1 など）の指定はできません。
HiRDB 運用ディレクトリの指定と同様に，［...］をクリックして，［フォルダの参照］ダイアログボックスからディレクトリを選択することもできます。

   注※
   %PDDIR%¥pdistup¥bin 下から簡易セットアップツールを起動しなかった場合は，空になります。
4. ［標準セットアップ］をクリックすると，セットアップが開始されます。

## 3.3.2 標準セットアップで設定される環境

標準セットアップを実行すると設定される環境を次に示します。

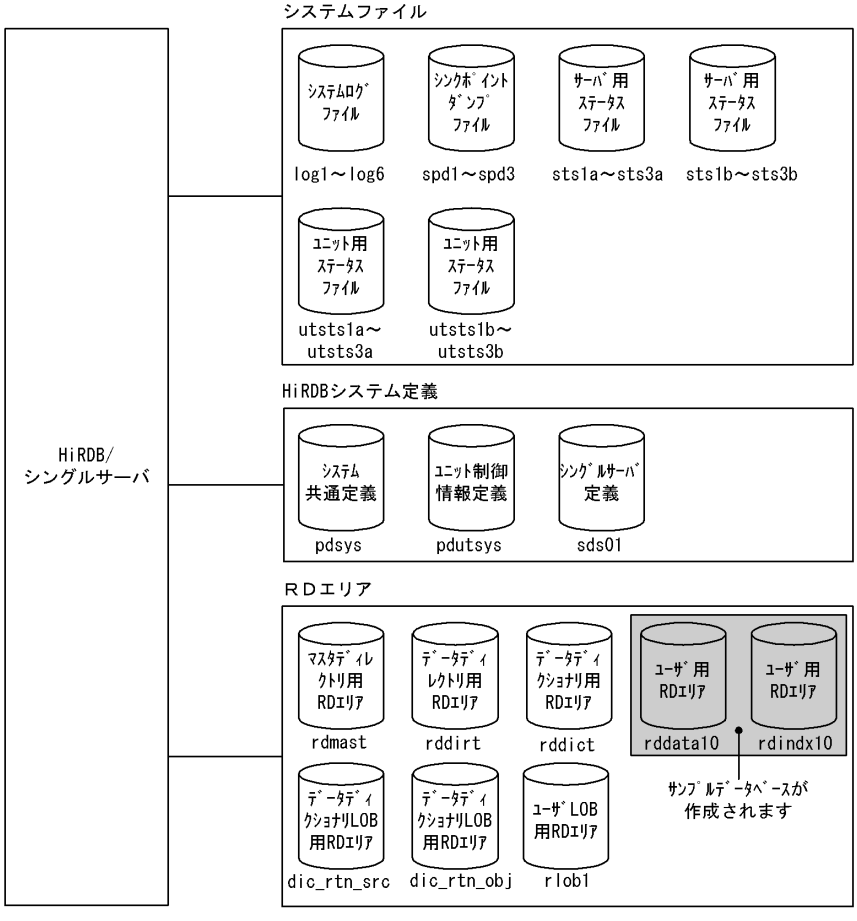
● 標準セットアップを実行すると登録される DBA 権限を持つ認可識別子
- ホストが UNIX の場合
  認可識別子：［hosts ファイルからホスト名を選択］ダイアログボックスの［ユーザ名］に指定した，HiRDB 管理者用のユーザ名
  パスワード：［hosts ファイルからホスト名を選択］ダイアログボックスで指定したパスワード
- ホストが Windows の場合
  認可識別子：root
  パスワード：root

● サンプルデータベースの作成
作成されます。

### （1） HiRDB/ シングルサーバの場合

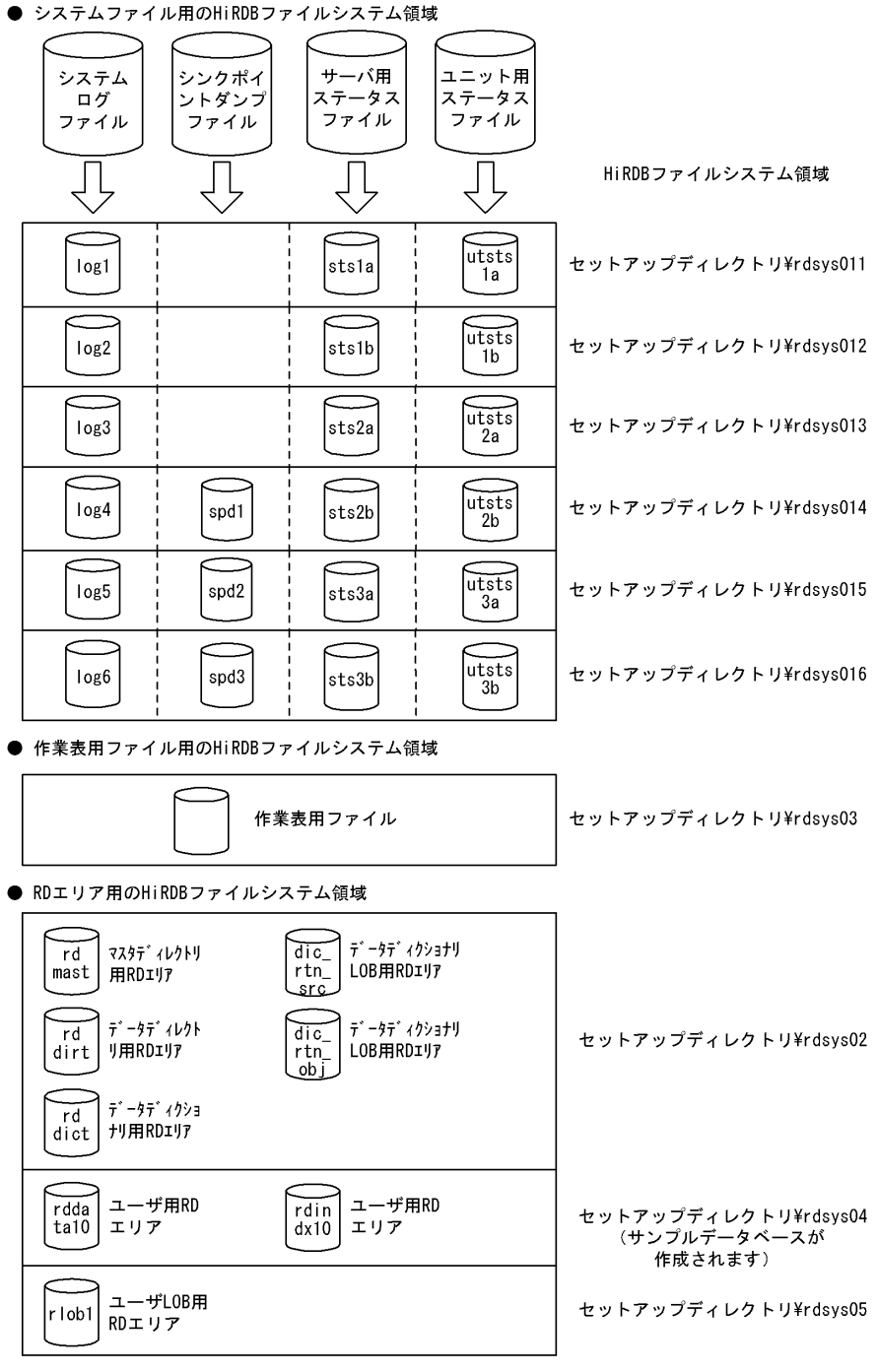
標準セットアップで作成されるシステム構成を次の図に示します。

**図 3-5　標準セットアップで作成されるシステム構成（HiRDB/ シングルサーバ）**




標準セットアップで作成される HiRDB ファイルシステム領域の構成を次の図に示します。HiRDB ファイルシステム領域は，セットアップディレクトリ下に作成されます。

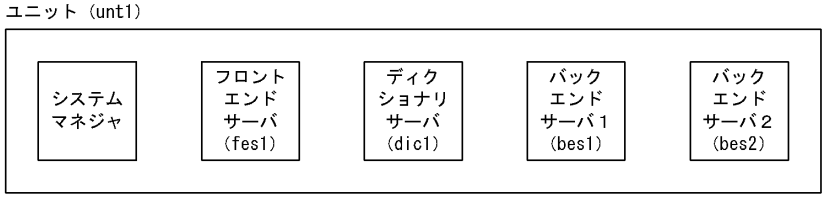
**図 3-6 標準セットアップで作成される HiRDB ファイルシステム領域の構成（HiRDB/シングルサーバ）**




## (2) HiRDB/パラレルサーバの場合

標準セットアップで作成されるシステム構成を次の図に示します。

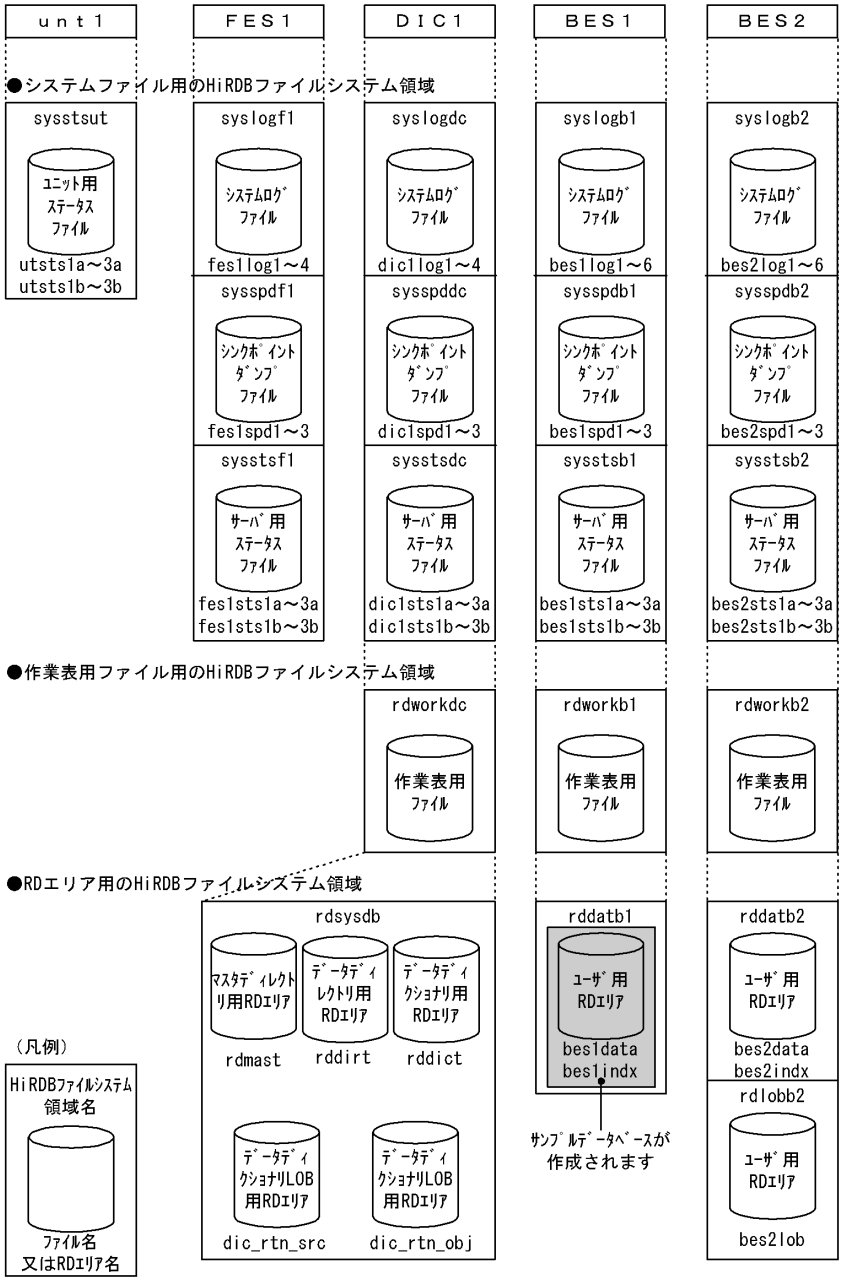
**図 3-7　標準セットアップで作成されるシステム構成（HiRDB/ パラレルサーバ）**




標準セットアップでは，簡単に HiRDB のシステムを構築するために，サーバマシンは 1 台だけという構成で環境設定します。HiRDB/ パラレルサーバは，複数のサーバマシンで動作し，データの検索などの処理を並列に実行できるシステムのため，「3.10　HiRDB/ パラレルサーバの環境設定をする場合」を参照して，カスタマイズしてください。

標準セットアップで作成される HiRDB ファイルシステム領域の構成を次の図に示します。HiRDB ファイルシステム領域は，セットアップディレクトリ下に作成されます。

**図 3-8 標準セットアップで作成される HiRDB ファイルシステム領域の構成（HiRDB/ パラレルサーバ）**




## 3.3.3 簡易セットアップツールで作成するサンプルデータベース

次に示す場合にサンプルデータベースが作成されます。

- 標準セットアップを実行した場合
- ウィザードセットアップを実行した場合※

- カスタムセットアップで［サンプルデータベースの作成］を選択して実行した場合※

注※
詳細定義画面で系切り替え機能の設定を行った場合，サンプルデータベースは作成されません。

この場合，サンプルデータベースとして次に示す表が作成されます。

**表名：**
CUSTOM，GOODS，VENDOR，TAKEODR，STOCK，WAREHUS，SHIPMNT，SENDODR，LAYIN

このうち，CUSTOM，GOODS，VENDOR，及び STOCK 表には，サンプルのデータが格納されます。

また，このとき，HiRDB に認可識別子が USER1（パスワードは USER1）のユーザが登録され，簡易セットアップツールを実行したマシンの HiRDB.ini ファイル中の PDUSER に，この認可識別子とパスワード（PDUSER=USER1/USER1）が自動的に設定されます。

# 3.4 ウィザードセットアップ

ウィザードセットアップでは，次に示す項目を対話形式で指定するだけで環境設定ができます。

- システム構成
- ユニット構成
- システムログの運用方法

## 3.4.1 ウィザードセットアップの場合の環境設定手順

ウィザードセットアップの場合の環境設定手順を次に示します。

### (1) システム構成を設定します

［HiRDB セットアップツール一開始］ウィンドウで，［ウィザードセットアップ］を選択すると，［HiRDB セットアップツール一ウィザードセットアップ［システム構成］］ウィンドウが表示されます。

HiRDB のシステム構成を設定します。

このウィンドウで設定する項目について説明します。

| 項目名 | 説明 |
|---|---|
| HiRDB **識別子** | 環境設定をする HiRDB の識別子を指定します。<br>4 文字で指定します。初期値は［HRD1］です。 |
| **ポート番号** | 環境設定をする HiRDB が使用するポート番号を指定します。<br>5001 ～ 65535 の範囲で指定します。初期値は 22200 です。 |
| **ユニット数** | HiRDB システムで構築するユニット数を指定します。HiRDB/ パラレルサーバの場合に指定できます。<br>1 ～ 999 の範囲で指定します。初期値は 1 です。 |
| **最大同時接続ユーザ数** | 最大同時接続ユーザ数を指定します。<br>1 ～ pd_max_users オペランドの上限値の範囲で指定します。初期値は 8 です。<br>なお，最大同時接続ユーザ数を変更した場合は，HiRDB の規模が再設定されます。 |

| 項目名 | 説明 |
|---|---|
| HiRDB の規模 | HiRDB の規模を，全ユニットで使用するディスク容量のサイズで指定します。<br>初期値～ 9,999,999 ギガバイトの範囲で指定します。初期値は，HiRDB/ シングルサーバの場合は 490 メガバイト，HiRDB/ パラレルサーバの場合は 550 メガバイトになります。<br>ユニット数，又は最大同時接続ユーザ数を変更した場合は，1 サーバ当たりに必要な RD エリア容量×サーバ台数（デフォルトの構成として，1 ユニット当たりフロントエンドサーバ 1 台，バックエンドサーバ 2 台を仮定），及びその RD エリア容量に必要なシステムファイル容量を加算した値が再設定されます。<br>なお，HiRDB 開始後の運用によっては，規模が拡大する場合があります。その場合の詳細については，「3.6.2(4)(a)　HiRDB ファイルシステム領域を編集します」の注を参照してください。 |
| RD エリア容量の割合 | HiRDB の規模で指定した容量のうち，RD エリアに使用する領域の割合を指定します。指定できる範囲，及び初期値は，HiRDB の規模によって異なります。 |

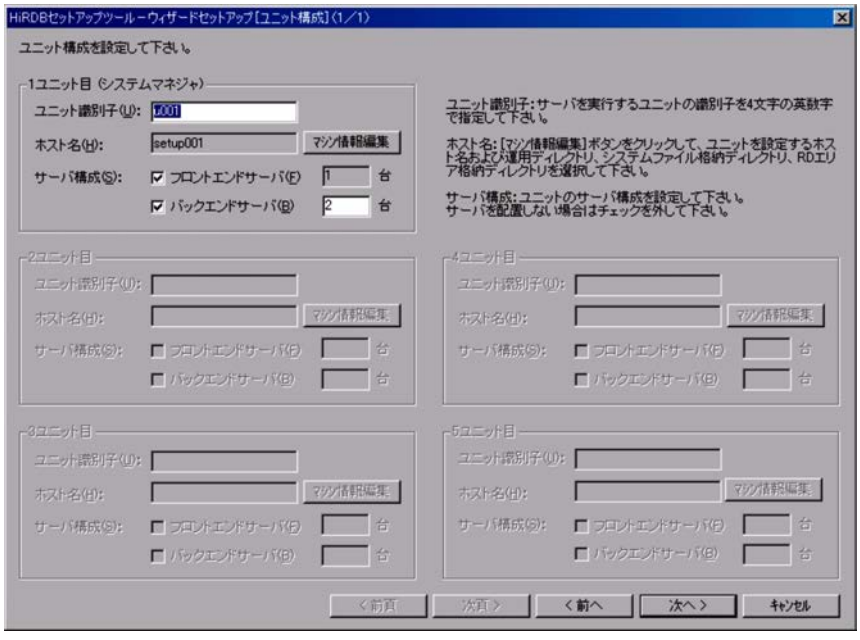
［次へ］をクリックすると，設定内容が保存され，HiRDB/ シングルサーバの場合は［HiRDB セットアップツール－ウィザードセットアップ［システムログ運用方法］］ウィンドウ，HiRDB/ パラレルサーバの場合は［HiRDB セットアップツール－ウィザードセットアップ［ユニット構成］］ウィンドウが表示されます。

なお，HiRDB の規模が十分な大きさでも，データベースの構成（サーバ台数や最大同時接続ユーザ数）によって RD エリア容量の割合が適切ではない場合は，次のとおり最適な RD エリア容量に自動調整されます。

- RD エリアの容量が不足する場合は，最小 RD エリア容量に自動調整します。
- システムファイル容量が不足する場合は，最大 RD エリア容量に自動調整します。

## (2) ユニット構成を設定します

HiRDB/ パラレルサーバの場合に，［HiRDB セットアップツール－ウィザードセットアップ［システム構成］］ウィンドウで［次へ］をクリックすると，［HiRDB セットアップツール－ウィザードセットアップ［ユニット構成］］ウィンドウが表示されます。




HiRDB/ パラレルサーバのユニット構成を設定します。［HiRDB セットアップツール－ウィザードセットアップ［システム構成］］ウィンドウで指定したユニット数分のユニット構成を設定します。

このウィンドウで設定する項目について説明します。

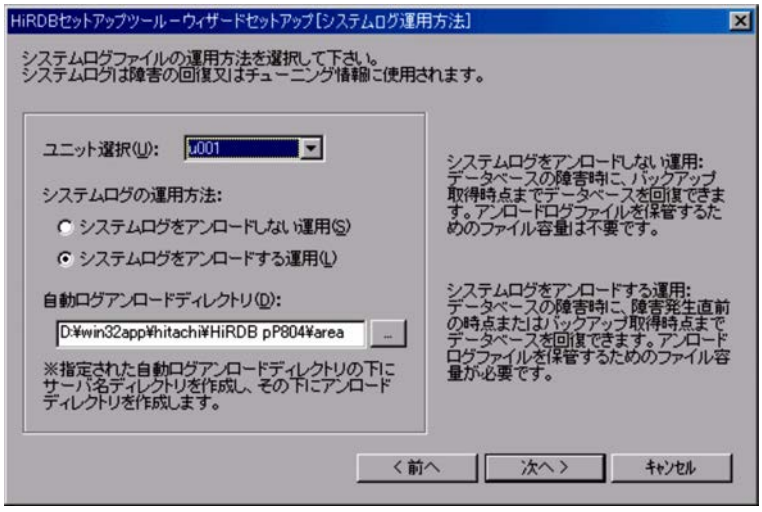
| 項目名 | 説明 |
|---|---|
| **ユニット識別子** | 設定するユニットの識別子を指定します。<br>4 文字で指定します。初期値は［u001］です。［HiRDB セットアップツール－ウィザードセットアップ［システム構成］］ウィンドウでユニット数に 2 以上を指定した場合，二つ目のユニットの初期値は［u002］となります。 |
| **ホスト名** | ［マシン情報編集］ボタンで［マシン情報編集］ダイアログボックスを表示し，サーバマシンの情報を設定します。［マシン情報編集］ダイアログボックスについては，「3.6.2(3)(a) マシン情報の編集」を参照してください。 |
| **フロントエンドサーバ** | フロントエンドサーバを設置する場合にチェックします。チェックがない場合は，サーバ定義を作成しません。なお，台数の編集はできません。 |
| **バックエンドサーバ** | バックエンドサーバを設置する場合にチェックします。チェックがない場合はサーバ定義を作成しません。また，ユニットに設置するバックエンドサーバの台数を指定します。<br>バックエンドサーバの台数は 0 ～ 34 の範囲で指定します。初期値は 2 です。<br>＜注意事項＞<br>バックエンドサーバの台数を増やしたときは，次のことに注意してください。<br>• 簡易セットアップツールで指定できるすべてのユニットのバックエンドサーバの合計台数は，1,998 台です。<br>• リソース数に関連する環境変数の設定が必要になる場合があります。リソース数に関連する環境変数の見積もりについては，「20. リソース数に関連する環境変数の見積もり」を参照してください。<br>• ユニット用ステータスファイルが不足することがあります。その場合は，［HiRDB セットアップツール－ウィザードセットアップ（システム構成）］ウィンドウで RD エリア容量の割合を調整するか，又は［HiRDB セットアップツール－詳細定義（セットアップ）］ウィンドウでユニット用ステータスファイルの容量を再設定してください。<br>• HiRDB ファイルシステム領域の見積もり計算でエラーが発生する場合があります。この場合は，エラーメッセージを参照して対策してください。 |

［次へ］をクリックすると，設定内容を保存し，［HiRDB セットアップツール－ウィザードセットアップ［システムログ運用方法］］ウィンドウを表示します。

### (3) システムログファイルの運用方法を設定します

HiRDB/ シングルサーバの場合は，［HiRDB セットアップツール－ウィザードセットアップ［システム構成］］ウィンドウで［次へ］をクリックすると，［HiRDB セットアップツール－ウィザードセットアップ［システムログ運用方法］］ウィンドウが表示されます。

HiRDB/ パラレルサーバの場合は，［HiRDB セットアップツール－ウィザードセットアップ［ユニット構成］］ウィンドウで［次へ］をクリックすると，［HiRDB セットアップツール－ウィザードセットアップ［システムログ運用方法］］ウィンドウが表示されます。




システムログファイルの運用方法を設定します。

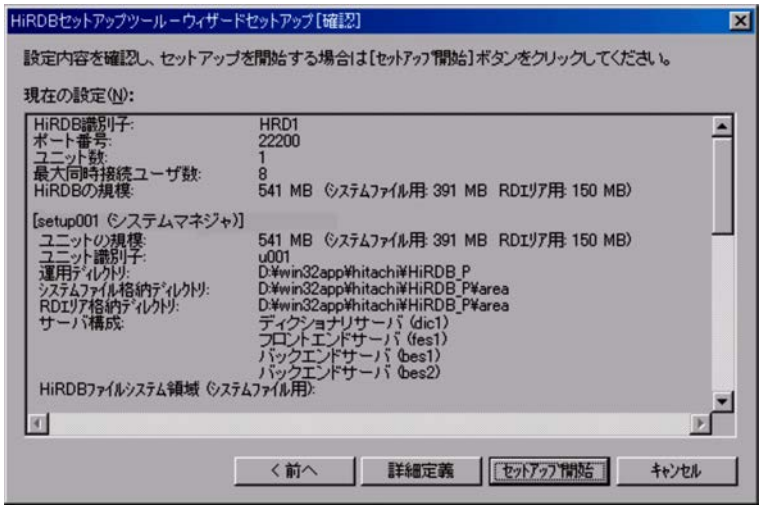
このウィンドウで設定する項目について説明します。

| 項目名 | 説明 |
|---|---|
| **ユニット選択** | 自動ログアンロードディレクトリを設定するユニットを選択します。 |
| **システムログの運用方法** | システムログの運用方法を選択します。初期値は［システムログをアンロードする運用］です。 |
| **自動ログアンロードディレクトリ** | 自動ログアンロードディレクトリを設定します。初期値はユニット構成で設定したシステムファイル格納ディレクトリ名を表示します。<br>システムログの運用方法で［システムログをアンロードする運用］を選択した場合に指定できます。 |

［次へ］をクリックすると，設定内容が保存され，［HiRDB セットアップツール－ウィザードセットアップ［確認］］ウィンドウが表示されます。

### (4) 設定内容を確認します

［HiRDB セットアップツール－ウィザードセットアップ［システムログ運用方法］］ウィンドウでシステムログファイルの運用方法の設定が完了すると［HiRDB セットアップツール－ウィザードセットアップ［確認］］ウィンドウが表示されます。




ウィザードセットアップでの設定内容，及び HiRDB ファイルシステム領域の割り当て容量（システムファイル用と RD エリア用の容量）の計算結果を表示します。

HiRDB ファイルシステム領域のシステムファイル用の容量は，ステータスファイルの容量見積もりの計算結果，及びシンクポイントダンプファイルの容量見積もりの計算結果を反映し，ユーザが指定したシステムファイル容量に収まるように割り当てられます。

［詳細定義］をクリックすると，［HiRDB セットアップツール―詳細定義（セットアップ）］ウィンドウが表示されます。カスタムセットアップ（詳細定義）については，「3.6　カスタムセットアップ（詳細定義）」を参照してください。

［セットアップ開始］をクリックすると，セットアップが開始されます。

# 3.5 カスタムセットアップ

カスタムセットアップでは，次に示す項目の指定だけで環境設定ができます。

- HiRDB の規模
- HiRDB 識別子
- ポート番号
- ユニット識別子
- システムファイル及び RD エリアの作成ディレクトリ

## 3.5.1 カスタムセットアップの場合の環境設定手順

カスタムセットアップの場合の環境設定手順を次に示します。

1. ［HiRDB セットアップツール－開始］ウィンドウで［カスタムセットアップ］をクリックすると，［HiRDB セットアップツール－カスタム］ウィンドウが表示されます。

2. HiRDB の規模，HiRDB 定義，セットアップディレクトリ，サンプルデータベースの作成有無を指定します。それぞれの項目について説明します。

| 項目名 | 説明 |
|---|---|
| **HiRDB の規模** | HiRDB の規模を指定します。初期値は［小規模］です。<br>**小規模**<br>約 500 メガバイトのディスク容量を使用する HiRDB 環境を作成します。<br>**中規模**<br>約 1 ギガバイトのディスク容量を使用する HiRDB 環境を作成します。<br>**大規模**<br>約 2 ギガバイトのディスク容量を使用する HiRDB 環境を作成します。<br>詳細定義画面で変更が行われると不活性になり，その後は編集できなくなります。また，定義情報の読み込みを実行すると不活性となり，その後は編集できなくなります。<br>HiRDB 開始後の運用によっては，規模が拡大する場合があります。その場合の詳細については，「3.6.2(4)(a)　HiRDB ファイルシステム領域を編集します」の注を参照してください。 |

| 項目名 | 説明 |
|---|---|
| HiRDB **定義** | 環境設定をする HiRDB の識別子，使用するポート番号，ユニット識別子を指定します。<br>HiRDB **識別子**<br>４文字で指定します。初期値は［HRD1］です。<br>**ポート番号**<br>5001 ～ 65535 の範囲で指定します。初期値は 22200 です。<br>**ユニット識別子**<br>４文字で指定します。初期値は［unt1］です。 |
| **セットアップディレクトリ** | システムファイルと RD エリアの格納先を分ける場合に指定します。初期値は［HiRDB セットアップツール一開始］ウィンドウの［セットアップディレクトリ］指定値です。<br>**システムファイル格納先**<br>システムファイルを作成するディレクトリを指定します。<br>RD **エリア格納先**<br>RD エリア用のファイルを作成するディレクトリを指定します。<br><br>なお，HiRDB/ パラレルサーバの場合，ここで指定する格納先はシステムマネジャがあるサーバです。システムマネジャがあるサーバ以外のサーバの格納先を指定する場合は，［マシン情報編集］ダイアログボックスで指定してください。［マシン情報編集］ダイアログボックスについては，「3.6.2(3)(a) マシン情報の編集」を参照してください。 |
| **サンプルデータベースの作成** | サンプルデータベースを作成するかどうかを指定します。 |

**注意事項**

ポート番号の指定で，次の場合はエラーメッセージが表示されます。

- 範囲外の値が指定された場合
- 指定したポート番号が別プログラムで使用されている場合
  別プログラムで使用されているポート番号とは，ホストが UNIX の場合，/etc/services ファイルに指定されているポート番号です。ホストが Windows の場合，%windir%¥system¥etc¥drivers¥services ファイルに指定されているポート番号です。
  ただし，HiRDB サーバ間での重複チェックはしません。

3.［セットアップ開始］をクリックすると，セットアップが開始されます。

# 3.6　カスタムセットアップ（詳細定義）

カスタムセットアップで詳細定義をすることによって，次の項目を設定できます。

- サーバマシンの情報の定義
- ユニット制御情報定義の編集
  - HiRDB ファイルシステム領域の指定
  - RD エリアの指定
  - ユニット用ステータスファイルの指定
- シングルサーバ定義，又は各サーバ定義の編集
  - サーバ用ステータスファイルの指定
  - システムログファイルの指定
  - シンクポイントダンプファイルの指定
- グローバルバッファの定義
- 系切り替え機能の設定
- リソース所要量の計算

## 3.6.1　メニュー一覧

カスタムセットアップで詳細定義する場合，［HiRDB セットアップツール－詳細定義（セットアップ）］ウィンドウで指定します。また，ウィザードセットアップで環境設定をした後に，［HiRDB セットアップツール－詳細定義（セットアップ）］ウィンドウで，より詳細に環境設定を行うこともできます。ウィザードセットアップについては，「3.4　ウィザードセットアップ」を参照してください。

［HiRDB セットアップツール－詳細定義（セットアップ）］ウィンドウを次の図に示します。また，このウィンドウのメニュー一覧を表 3-2 に示します。

**図 3-9 ［HiRDB セットアップツール－詳細定義（セットアップ）］ウィンドウ**

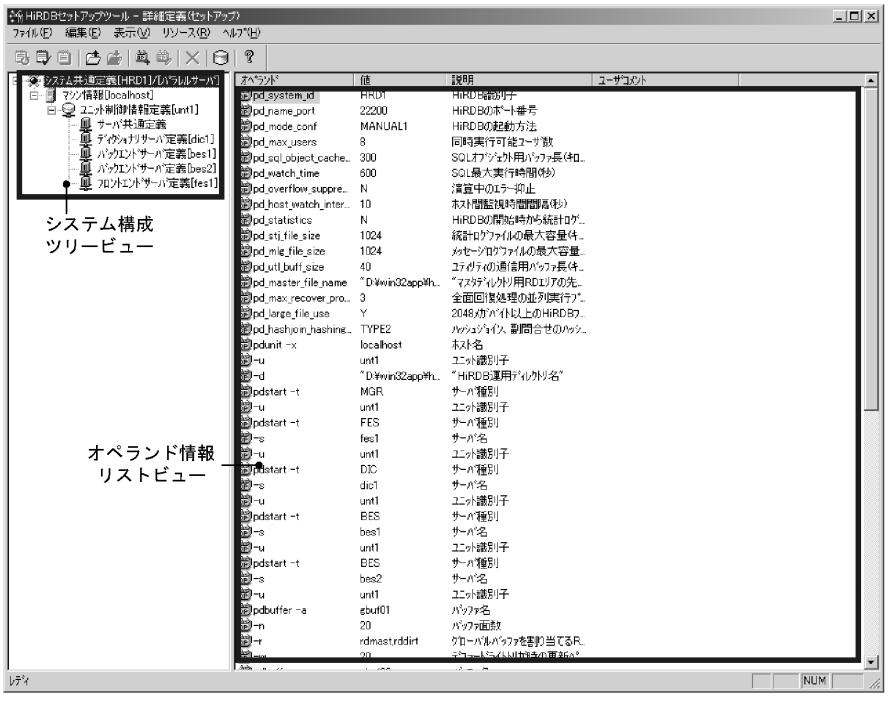



**表 3-2 ［HiRDB セットアップツール－詳細定義（セットアップ）］ウィンドウのメニュー一覧**

| メニュー | ドロップダウンメニュー | 説明 |
|---|---|---|
| ファイル（F） | システム定義の更新（U）... | HiRDB のシステム定義を更新します。定義更新時だけ選択できます。<br>システム定義のチェックで問題ない場合だけ更新できます。チェックで問題がある場合は選択できません。 |
| | システム定義のチェック（C）... | システム定義の内容をチェックします（pdconfchk コマンドの実行）。 |
| | システム定義のチェック結果（R）... | 前回のシステム定義のチェック結果を表示します。システム定義が一度もチェックされていない場合は選択できません。 |
| | 詳細定義情報の読み込み（L） | ローカルディスク上に単一ファイル形式，又はディレクトリ構造形式で保存した詳細定義情報を読み込みます。定義更新時は選択できません。 |
| | 詳細定義情報の上書き保存（S）※9 | 詳細定義情報を上書き保存します。一度も詳細定義情報の保存をしていない場合は，名前を付けて保存します。システム定義のチェックで問題ない場合だけ保存できます。チェックで問題がある場合，及び定義更新時は選択できません。 |
| | 詳細定義情報の保存（A）※9 | ローカルディスク上に，詳細定義情報を名前を付けて保存します。システム定義のチェックで問題ない場合だけ保存できます。チェックで問題がある場合，及び定義更新時は選択できません。 |

| メニュー | ドロップダウンメニュー | | 説明 |
|---|---|---|---|
| | 終了（X） | | 詳細定義を終了します。 |
| 編集（E） | 追加（A）※1 | マシン（M）※2 | システム構成ツリービューにサーバマシンを追加します（「3.10(1) サーバマシンを追加します」参照）。マシンを追加すると，ユニット制御情報定義，サーバ共通定義，及びバックエンドサーバ定義も追加されます。 |
| | | ユニット（U）※3 | システム構成ツリービューにユニット制御情報定義を追加します。ユニットを追加すると，サーバ共通定義及びバックエンドサーバ定義も追加されます。 |
| | | サーバ共通定義（C）※4 | システム構成ツリービューにサーバ共通定義を追加します。 |
| | | ディクショナリサーバ（D）※4 | システム構成ツリービューにディクショナリサーバ定義を追加します。 |
| | | フロントエンドサーバ（F）※4 | システム構成ツリービューにフロントエンドサーバ定義を追加します。 |
| | | バックエンドサーバ（B）※4 | システム構成ツリービューにバックエンドサーバ定義を追加します。 |
| | | オペランド（O）... | ［追加オペランド選択］ダイアログボックスを表示します（「3.6.2(6)(a) オペランド又はオプションフラグの追加」参照）。 |
| | | オプションフラグ（P）...※5 | ［追加オプションフラグ選択］ダイアログボックスを表示します「3.6.2(6)(a) オペランド又はオプションフラグの追加」参照）。 |
| | 編集（E）※6 | HiRDB ファイルシステム領域（H）...※4 | ［HiRDB ファイルシステム領域編集］ダイアログボックスを表示します（「3.6.2(4)(a)HiRDB ファイルシステム領域を編集します」参照）。 |
| | | RD エリア（R）...※4 | ［RD エリア編集］ダイアログボックスを表示します（「3.6.2(4)(b)RD エリアを編集します」参照）。 |
| | | ステータスファイル（S）...※4，※7 | ［ステータスファイル編集］ダイアログボックスを表示します（ユニット用ステータスファイルの場合は「3.6.2(4)(d) ユニット用ステータスファイルを編集します」参照，サーバ用ステータスファイルの場合は「3.6.2(5)(a) サーバ用ステータスファイルを編集します」参照）。 |
| | | システムログファイル（L）...※7 | ［システムログファイル編集］ダイアログボックスを表示します（「3.6.2(5)(b) システムログファイル及びシンクポイントダンプファイルを編集します」参照）。 |
| | | シンクポイントダンプファイル（D）...※7 | ［システムログファイル編集］ダイアログボックスを表示します（「3.6.2(5)(b) システムログファイル及びシンクポイントダンプファイルを編集します」参照）。 |
| | | グローバルバッファ（G）...※2 | ［グローバルバッファ編集］ダイアログボックスを表示します（「3.6.2(7) グローバルバッファを定義します」参照）。 |
| | | マシン情報（M）...※3 | ［マシン情報編集］ダイアログボックスを表示します（「3.6.2(3)(a) マシン情報の編集」参照）。 |

| メニュー | ドロップダウンメニュー | | 説明 |
|---|---|---|---|
| | | 系切り替え機能の設定（A）... ※2 | ［系切り替え機能の設定］ダイアログボックスを表示します（「3.11　系切り替え機能の設定」参照）。 |
| | | サーバ名（N）... ※7 | ［サーバ名設定］ダイアログボックスを表示します（「3.6.2(3)(b) サーバ名の設定」参照）。 |
| | | オペランド（O）... | オペランドの形式に応じた［オペランド値編集］ダイアログボックスを表示します（「3.6.2(6)(b) オペランド値の編集」参照）。 |
| | 削除（D） | | 選択した項目を削除します。 |
| | コピー（C） | | 選択した項目をコピーしてバッファに保存します。 |
| | 貼り付け（P） | | バッファにコピーした内容を貼り付けます。 |
| 表示（V） | ツールバー（T） | | ツールバーの表示，又は非表示を指定します。 |
| | ステータスバー（S） | | ステータスバーの表示，又は非表示を指定します。 |
| リソース（R）※8 | リソース所要量の計算（C） | | ［リソース所要量結果］ダイアログボックスを表示します（「3.6.2(9) リソース所要量を確認します」参照）。 |
| ヘルプ（H） | バージョン情報（A）... | | ［バージョン情報］ダイアログボックスを表示します。 |

注※1
HiRDB/ シングルサーバの環境設定時，及び定義更新時は［オペランド］以外は選択できません。

注※2
システム構成ツリービューで［システム共通定義］を選択している場合にだけ，選択できます。

注※3
システム構成ツリービューで［マシン情報］を選択している場合にだけ，選択できます。

注※4
システム構成ツリービューで［ユニット制御情報定義］を選択している場合にだけ，選択できます。

注※5
オペランド情報リストビューでコマンド形式オペランドを選択している場合にだけ，選択できます。

注※6
定義更新時は［系切り替え機能の設定］及び［オペランド］以外は選択できません。

注※7
システム構成ツリービューで各サーバ（［サーバ共通定義］を除く）を選択している場合にだけ，選択できます。

注※8
簡易セットアップツールで環境設定していない環境で定義更新をした場合，リソース所要量の計算に必要なファイルがないため，選択できません。

注※9
詳細定義情報は単一ファイル形式，又はディレクトリ構造形式でローカルディスク上に保存します。このとき，ホスト名やパス名を変数に置き換えて保存します。読み込み時は，それらの変数を実際の

ホスト名やパス名に置き換えて読み込みます。保存時，及び読み込み時に置き換える変数名を次に示します。

| 変数名 | 保存時 | 読み込み時 |
|---|---|---|
| ?MASTDIR? | マスタディレクトリ用 RD エリアの先頭の HiRDB ファイル名が RD エリア格納先ディレクトリで始まっている場合に置き換えます。 | **[HiRDB セットアップツール－開始] ウィンドウから読み込んだ場合：**[HiRDB セットアップツール－開始] ウィンドウで設定したセットアップディレクトリと，読み込みファイル内のマスタディレクトリ用 RD エリアの先頭の HiRDB ファイル名を連結した値に置き換えます。<br>**[HiRDB セットアップツール－詳細定義（セットアップ）] ウィンドウからから読み込んだ場合：**RD エリア格納先ディレクトリと，読み込む定義ファイル内のマスタディレクトリ用 RD エリアの先頭の HiRDB ファイル名を連結した値に置き換えます。 |
| ?HOST? | [ マシン情報編集 ] ウィンドウの [ ホスト名 ] に指定した文字列を変数に置き換えます。 | [HiRDB セットアップツール－開始] ウィンドウで設定したホスト名に置き換えます。<br>HiRDB/ パラレルサーバで複数のサーバマシンを指定した場合は，保存時に設定したホスト名に置き換えます。それ以外のホスト名は HOST1，HOST2，のように仮のホスト名に置き換えます。仮のホスト名については，詳細定義情報を読み込んだ後で [ マシン情報編集 ] ウィンドウの [ ホスト名 ]，[HiRDB 運用ディレクトリ ]，[ システムファイル格納先 ]，及び [RD エリア格納先 ] を設定し直してください。 |
| ?HIDIR? | [ マシン情報編集 ] の [HiRDB 運用ディレクトリ ] に指定した文字列を変数に置き換えます。 | [HiRDB セットアップツール－開始] ウィンドウで設定した運用ディレクトリに置き換えます。 |
| ?RDDIR? | [ マシン情報編集 ] の [ システムファイル格納先 ] に指定した文字列を変数に置き換えます。 | **[HiRDB セットアップツール－開始] ウィンドウから読み込んだ場合：**開始ウィンドウで設定したセットアップディレクトリに置き換えます。<br>**[HiRDB セットアップツール－詳細定義（セットアップ）] ウィンドウから読み込んだ場合：**RD エリア格納先ディレクトリに置き換えます。 |
| ?SYSDIRA? | [ マシン情報編集 ] の [RD エリア格納先 ] に指定した文字列を変数に置き換えます。 | **[HiRDB セットアップツール－開始] ウィンドウから読み込んだ場合：**開始ウィンドウで設定したセットアップディレクトリに置き換えます。<br>**[HiRDB セットアップツール－詳細定義（セットアップ）] ウィンドウから読み込んだ場合：**システムファイル格納先ディレクトリに置き換えます。 |
| ?SYSDIRB? | 置き換えません。 | 置き換えません。 |

## 3.6.2 カスタムセットアップ（詳細定義）の場合の環境設定手順

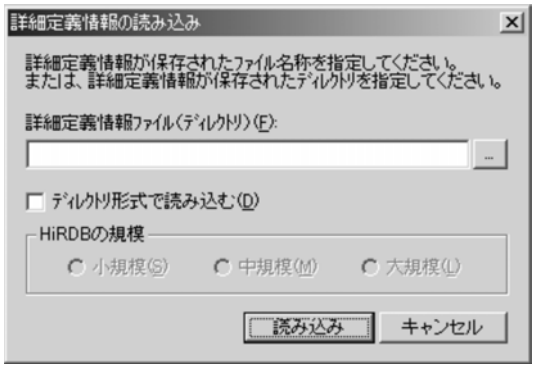
カスタムセットアップ（詳細定義）の場合の環境設定手順を次に示します。

### (1) [HiRDB セットアップツール－詳細定義（セットアップ）] ウィンドウを表示させます

「3.5.1 カスタムセットアップの場合の環境設定手順」の手順 1. ～ 2. を実行して，［詳細定義］をクリックします。［HiRDB セットアップツール－詳細定義（セットアップ）］ウィンドウが表示されます。

### (2) 詳細定義情報を読み込みます

ローカルディスク上に保存した詳細定義情報を読み込む場合は，メニューの［ファイル］－［詳細定義情報の読み込み］を選択します。［詳細定義情報の読み込み］ダイアログボックスが表示されます。




保存した詳細定義情報ファイル（ディレクトリ）を選択します。

このダイアログボックスで指定する項目について説明します。

| 項目名 | 説明 |
|---|---|
| **詳細定義情報ファイル（ディレクトリ）** | 読み込む詳細定義情報ファイル又はディレクトリをフルパスで指定します。<br>ディレクトリ構造形式で保存した場合は指定ディレクトリ直下に“conf”及び“ini”ディレクトリが作成されますが，詳細定義情報の保存で指定したディレクトリ名を入力又は選択してください。 |
| **ディレクトリ形式で読み込む** | 詳細定義情報をディレクトリ形式で読み込む場合にチェックします。 |
| HiRDB **の規模** | ［ディレクトリ形式で読み込む］をチェックしている場合に選択できます。指定した規模の定義情報が保存されていない場合はエラーになります。 |

［読み込み］をクリックすると，詳細定義情報を読み込み，［HiRDB セットアップツール－詳細定義（セットアップ）］ウィンドウに戻ります。

**注意事項**

HiRDB/ パラレルサーバで複数のサーバマシンを指定した場合は，保存時に設定したホスト名に置き換えます。それ以外のホスト名は HOST1，HOST2，のように仮のホスト名に置き換えます。仮のホスト名については，詳細定義情報を読み込んだ後で [ マシン情報編集 ] ウィンドウの [ ホスト名 ]，[HiRDB 運用ディレクトリ ]，[ システムファイル格納先 ]，及び [RD エリア格納先 ] を設定し直してください。

## (3) サーバマシンの情報を定義します

環境設定をする HiRDB がインストールされているサーバマシンの情報を定義します。なお，HiRDB/ パラレルサーバでサーバマシンを追加する場合は，「3.10　HiRDB/ パラレルサーバの環境設定をする場合」を参照してください。

### (a) マシン情報の編集

環境設定をする HiRDB がインストールされているマシンのホスト名，HiRDB 運用ディレクトリ，システムファイル及び RD エリアの格納先を編集します。

1. システム構成ツリービューで［マシン情報］を選択した状態で，メニューの［編集］－［編集］－［マシン情報］を選択すると，［マシン情報編集］ダイアログボックスが表示されます。

［ホスト名］には，［HiRDB セットアップツール－詳細定義（セットアップ）］ウィンドウのシステム構成ツリービューで選択しているマシンのホスト名が，ホスト名，FQDN，又は IP アドレスの形式で表示されます。

ここでは，システムマネージャノードの HiRDB 運用ディレクトリを変更しないでください。変更する場合は開始画面で変更してください。

2. 環境設定をするマシンのホスト名を変更する場合，［hosts ファイルからホスト名を選択］をクリックします。

［hosts ファイルからホスト名を選択］ダイアログボックスが表示されるので，ホスト名，Windows ディレクトリ，ユーザ名，及びパスワードを指定します。

| 項目名 | 説明 |
|---|---|
| **ホスト名** | 環境設定をするマシンのホスト名を指定します。［hosts ファイルのホスト名一覧］から選択するか，又は直接入力してください。入力できる形式は，ホスト名，FQDN，又は IP アドレスです。また，簡易セットアップツールを実行しているマシンの場合は，ループバックアドレスも指定できます。<br>なお，ホスト名を変更した場合，［ユーザ名］及び［パスワード］の内容はリセットされるため，入力し直してください。ただし，システムマネジャがあるサーバのホスト名は変更しないでください。 |
| Windows **ディレクトリ** | 環境設定をするマシンが Windows の場合は Windows インストールディレクトリをフルパスで指定します。通常は変更する必要はありません。セットアップ先が UNIX の場合は，指定は無視されます。<br>セパレータは ¥ と / のどちらも使用できますが，混在して指定できません。また，「**ドライブ名**:¥」と「/ **ドライブ名**」のどちらも指定できます。 |

| 項目名 | 説明 |
|---|---|
| **ユーザ名** | 環境設定をするサーバマシンの HiRDB 管理者用のユーザ名を指定します。ドメインユーザの場合はドメインを付けて指定する必要はありません。 |
| **パスワード** | ［ホスト名］で設定したホストの HiRDB 管理者のパスワードを指定します。<br>選択したホストが UNIX の場合，.rhosts に設定したパスワードを使用してログインするため，パスワードのチェックはしません。<br>選択したホストが Windows の場合，選択したホストにアクセスするとき，このパスワードを併用するため，選択したホストのログインユーザと一致させてください。 |

なお，簡易セットアップツールで環境設定をすると，HiRDB に次に示す認可識別子を登録して DBA 権限を付与します。

- ［ホスト名］で選択したホストが UNIX の場合
  ユーザ：［ユーザ名］に指定した HiRDB 管理者用のユーザ名
  パスワード：［パスワード］に指定したパスワード
- ［ホスト名］で選択したホストが Windows の場合
  ユーザ：root
  パスワード：root

また，この設定のリモートホスト接続処理で，セットアップ先にワーク領域を作成します。ワーク領域には，セットアップ先に正常に接続しているかを確認するためのバッチファイルや，バッチファイルの実行結果を保存したファイルを格納します。ワーク領域は，次の場所に作成されます。

- ［ホスト名］で選択したホストが UNIX の場合
  /tmp/[ ログインユーザ名 ]
- ［ホスト名］で選択したホストが Windows の場合
  %windir%¥Temp¥[ ログインユーザ名 ]

3. ［HiRDB 運用ディレクトリ］で HiRDB 運用ディレクトリを指定します。
   セパレータは ¥ と / のどちらも使用できますが，混在して指定できません。また，「**ドライブ名** :¥」と「/ **ドライブ名**」のどちらも指定できます。
   ただし，次の場合のセパレータは ¥ になります。
   - 簡易セットアップツールが作成するシステム定義ファイル中のパス
   - 簡易セットアップツールの画面に表示されるパス
4. ［システムファイル格納先］でシステムファイルの格納先ディレクトリを指定します。
   初期値は %PDDIR%¥area です。セパレータは ¥ と / のどちらも使用できますが，混在して指定できません。また，「**ドライブ名** :¥」と「/ **ドライブ名**」のどちらも指定できます。
5. ［RD エリア格納先］で RD エリアの格納先ディレクトリを指定します。
   初期値は %PDDIR%¥area です。セパレータは ¥ と / のどちらも使用できますが，混在して指定できません。また，「**ドライブ名** :¥」と「/ **ドライブ名**」のどちらも指定できます。
6. ［OK］をクリックすると，［HiRDB セットアップツール―詳細定義（セットアップ）］ウィンドウに戻ります。

### (b) サーバ名の設定

環境設定をする HiRDB がインストールされているマシンのサーバ名を設定します。

1. システム構成ツリービューで各サーバを選択した状態で，メニューの［編集］―［編集］―［サーバ名］を選択すると，［サーバ名設定］ダイアログボックスが表示されます。

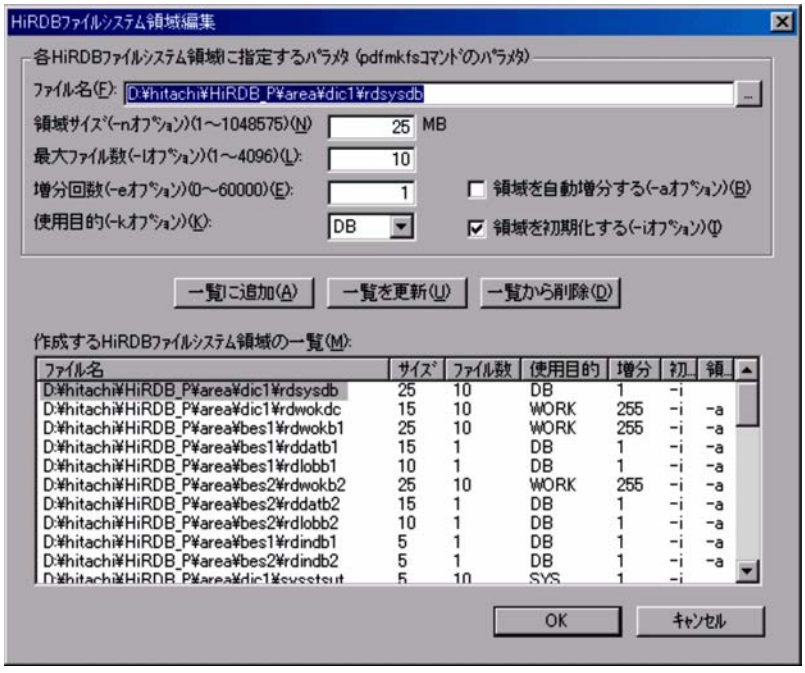
2. 変更後のサーバ名を 1 ～ 8 文字で指定します。
3. ［OK］をクリックすると，［HiRDB セットアップツール－詳細定義（セットアップ）］ウィンドウに戻ります。

## (4) ユニット制御情報定義を編集します

ユニット制御情報定義で定義する内容を編集します。

### (a) HiRDB ファイルシステム領域を編集します

システム構成ツリービューで［ユニット制御情報定義］を選択した状態で，メニューの［編集］－［編集］－［HiRDB ファイルシステム領域］を選択すると，［HiRDB ファイルシステム領域編集］ダイアログボックスが表示されます。




簡易セットアップツールで作成する HiRDB ファイルシステム領域を，［作成する HiRDB ファイルシステム領域の一覧］リストに［一覧に追加］で追加します。このダイアログボックス表示時は，初期値の HiRDB ファイルシステム領域が表示されるので，リストで選択して，［一覧を更新］又は［一覧から削除］で編集したり，削除したりできます。

このダイアログボックスで指定する項目について説明します。

| 項目名 | 説明 |
|---|---|
| **ファイル名** | 作成する HiRDB ファイルシステム領域名をフルパスで指定します。 |
| **領域サイズ** | HiRDB ファイルシステム領域として割り当てる容量をメガバイト単位で指定します。 |
| **最大ファイル数** | HiRDB ファイルシステム領域内に作成する HiRDB ファイル数の上限を指定します。 |
| **増分回数** | HiRDB ファイルシステム領域内の HiRDB ファイルの増分回数の上限値を指定します。 |
| **使用目的** | HiRDB ファイルシステム領域の使用目的を指定します。SDB は指定できません。また，系切り替え機能を使用する（pd_ha オペランドに use を指定）場合，SVR は指定できません。指定すると［OK］時にエラーになります。 |
| **領域を自動増分する** | HiRDB ファイルシステム領域の自動増分を行う場合に指定します。<br>使用目的が DB，SYS，又は WORK のときだけチェックできます。また，使用目的が DB，又は SYS のときにこの項目を指定すると，最大ファイル数と増分回数が変更できなくなります。<br>HiRDB の規模を拡大させたくない場合については，「3.1.1　簡易セットアップツールとは」の「注意」を参照してください。 |
| **領域を初期化する** | HiRDB ファイルシステム領域を最初から初期化する場合に指定します。系切り替え機能使用時は，この項目を必ずチェックしてください。チェックしないと［OK］時にエラーになります。 |

［HiRDB ファイルシステム領域編集］ダイアログボックスで［OK］をクリックすると，［HiRDB セットアップツール―詳細定義（セットアップ）］ウィンドウへ戻ります。

### (b) RD エリアを編集します

システム構成ツリービューで［ユニット制御情報定義］を選択した状態で，メニューの［編集］―［編集］―［RD エリア］を選択すると，［RD エリア編集］ダイアログボックスが表示されます。

簡易セットアップツールで作成する RD エリアを，［RD エリアの一覧］リストに［一覧に追加］で追加します。このダイアログボックス表示時は，初期値の RD エリアが表示されるので，リストで選択して，［一覧を更新］又は［一覧から削除］で編集したり，削除したりできます。また，作成する RD エリアに割り当てる HiRDB ファイルを［RD エリアを割り当てる HiRDB ファイルの指定］ダイアログボックスで指定します。

［RD エリア編集］ダイアログボックスで［OK］をクリックすると，［HiRDB セットアップツール―詳細定義（セットアップ）］ウィンドウへ戻ります。

**● RD エリアの編集**

［RD エリア編集］ダイアログボックスで指定する項目について説明します。

| 項目名 | 説明 |
|---|---|
| **RD エリア名** | 作成する RD エリア名を 1 ～ 30 文字で指定します。 |
| **RD エリアの種類** | RD エリアの種類を選択します。なお，次の指定はできません。<br>• shared<br>• datadictionary of routine<br>• user used by 認可識別子<br>• LOB used by HiRDB（任意のディクショナリ表）<br>• LOB used by 認可識別子 |
| **サーバ名** | RD エリアを管理するサーバを選択します。なお，HiRDB/ シングルサーバの場合は指定できません。 |
| **ページ長** | RD エリアを構成する HiRDB ファイルのページ長を 2,048 の倍数のバイト単位で指定します。 |
| **セグメントサイズ** | 1 セグメントの大きさをページ数で指定します。 |
| **最大リスト登録数** | リスト用 RD エリアを指定する場合の最大リスト登録数を指定します。シングルサーバ又はバックエンドサーバのときだけ指定できます。 |
| **自動増分を適用する** | 自動増分を適用するかどうかを指定します。ただし，次の RD エリアの場合は選択できません。<br>• マスタディレクトリ用 RD エリア<br>• データディレクトリ用 RD エリア |
| **増分セグメント数** | 自動増分を適用する場合の増分セグメント数を指定します。 |

**●作成した RD エリアに割り当てる HiRDB ファイルの指定**

［RD エリア編集］ダイアログボックスの［RD エリアに割り当てる HiRDB ファイルの指定］をクリックすると，［RD エリアを割り当てる HiRDB ファイルの指定］ダイアログボックスが表示されます。

作成する RD エリアに割り当てる HiRDB ファイルがある HiRDB ファイルシステム領域，HiRDB ファイル名（1 ～ 30 文字），及びその HiRDB のセグメント数を指定します。［OK］をクリックすると，［RD エリア編集］ダイアログボックスに戻ります。

### (c) HiRDB ファイルシステム領域及び RD エリアの編集時の注意事項

- raw I/O 機能を使用する場合，［HiRDB ファイルシステム領域編集］ダイアログボックスの［ファイル名］には，「¥¥.¥ **ドライブ名** :」の形式で指定してください。リモートで環境設定をする場合，［フォルダの参照］ダイアログボックスで，未フォーマットのパーティションの論理ドライブを選択できます。このときも［ファイル名］には「¥¥.¥ **ドライブ名** :」の形式で指定してください。
- raw I/O 機能を適用した HiRDB ファイルシステム領域には制限があります。制限の詳細については，「4.3.2(1)raw I/O 機能の適用範囲」を参照してください。

### (d) ユニット用ステータスファイルを編集します

簡易セットアップツールで作成するユニット用ステータスファイルは，［ステータスファイル編集］ダイアログボックスでリストに［一覧に追加］で追加します。このダイアログボックス表示時は，初期値の値が表示されるので，リストで選択して，［一覧を更新］又は［一覧から削除］で編集したり，削除したりできます。

ユニット用の［ステータスファイル編集］ダイアログボックスを表示するには，システム構成ツリービューで［ユニット制御情報定義］を選択した状態で，メニューの［編集］－［編集］－［ステータスファイル］を選択します。

作成できるステータスファイルは 7 個のため，［作成するステータスファイルの一覧］リストのファイルが 7 個になると，［一覧に追加］ボタンはクリックできなくなります。また，［作成するステータスファイルの一覧］リストのファイルが 3 個になると，［一覧から削除］ボタンはクリックできなくなります。

このダイアログボックスで指定する項目について説明します。

| 項目名 | 説明 |
|---|---|
| **論理ファイル名** | 作成するステータスファイルの論理ファイル名を英数字 1 ～ 8 文字で指定します。初期値を次に示します。<br>サーバの場合<br>　[ **サーバ名** ]sts[ **論理ファイル数の通番** ]<br>　ただし，8 文字を超える場合は次の名称とします。<br>　[ **サーバ種別**※ **（1 文字）** ][ **サーバの通番** ]s[ **論理ファイル数の通番** ]<br>　注※<br>　　d：ディクショナリサーバ<br>　　f：フロントエンドサーバ<br>　　b：バックエンドサーバ<br>ユニットの場合<br>　[ **ユニット名の 1，2，4 文字** ]sts[ **論理ファイル数の通番** ]<br>　ただし，8 文字を超える場合は次の名称とします。<br>　u[ **ユニットの通番** ]s[ **論理ファイル数の通番** ] |
| **A 系 HiRDB ファイルシステム領域名** | 作成する A 系ステータスファイルを格納する HiRDB ファイルシステム領域名をドロップダウンリストで選択します。種別が SYS と SVR 以外の HiRDB ファイルシステム領域は選択できません。初期値を次に示します。<br>サーバの場合<br>　[ **ファイルの格納先** ]¥[ **サーバ名** ]¥syssts[ **サーバの種類** ][ **サーバ数の通番** ]¥<br>ユニットの場合<br>　[ **ファイルの格納先** ]¥dic1¥sysstsut¥ |

| 項目名 | 説明 |
|---|---|
| **A 系ステータスファイル名** | 作成する A 系ステータスファイル名を英数字で指定します。初期値は「**[ サーバ名 ]**sts**[ 論理ファイル数の通番 ]**a」です。 |
| **B 系 HiRDB ファイルシステム領域名** | 作成する B 系ステータスファイルを格納する HiRDB ファイルシステム領域名をドロップダウンリストで選択します。種別が SYS と SVR 以外の HiRDB ファイルシステム領域は選択できません。初期値を次に示します。<br>サーバの場合<br>**[ ファイルの格納先 ]**¥**[ サーバ名 ]**¥sysst**s[ サーバの種類 ][ サーバ数の通番 ]**¥<br>ユニットの場合<br>**[ ファイルの格納先 ]**¥dic1¥sysstsut¥ |
| **B 系ステータスファイル名** | 作成する B 系ステータスファイル名を英数字で指定します。初期値は「**[ サーバ名 ]**sts**[ 論理ファイル数の通番 ]**b」です。 |
| **レコード長** | ステータスファイルのレコード長をバイト単位で指定します。 |
| **レコード数** | ステータスファイルのレコード数を指定します。 |

［ステータスファイル編集］ダイアログボックスで［OK］をクリックすると，［HiRDB セットアップツール－詳細定義（セットアップ）］ウィンドウへ戻ります。

## (5) シングルサーバ定義，又は各サーバ定義を編集します

簡易セットアップツールで作成するサーバ用ステータスファイル，システムログファイル，及びシンクポイントダンプファイルは，それぞれの編集ダイアログボックスでリストに［一覧に追加］で追加します。それぞれの編集ダイアログボックス表示時は，初期値の値が表示されるので，リストで選択して，［一覧を更新］又は［一覧から削除］で編集したり，削除したりできます。

それぞれのファイルについて説明します。

### (a) サーバ用ステータスファイルを編集します

サーバ用の［ステータスファイル編集］ダイアログボックスを表示するには，システム構成ツリービューで各サーバ※の定義を選択した状態で，メニューの［編集］－［編集］－［ステータスファイル］を選択します。

注※

シングルサーバ，ディクショナリサーバ，バックエンドサーバ，及びフロントエンドサーバのことです。

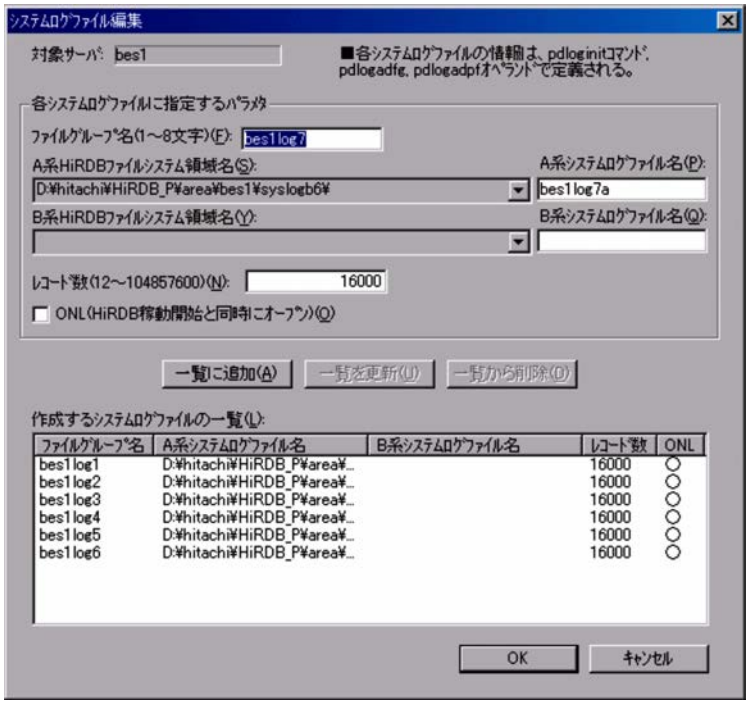
サーバ用の［ステータスファイル編集］ダイアログボックスで指定する項目については，「3.6.2(4)(d) ユニット用ステータスファイルを編集します」を参照してください。

### (b) システムログファイル及びシンクポイントダンプファイルを編集します

編集ダイアログボックスを表示する手順は次のとおりです。

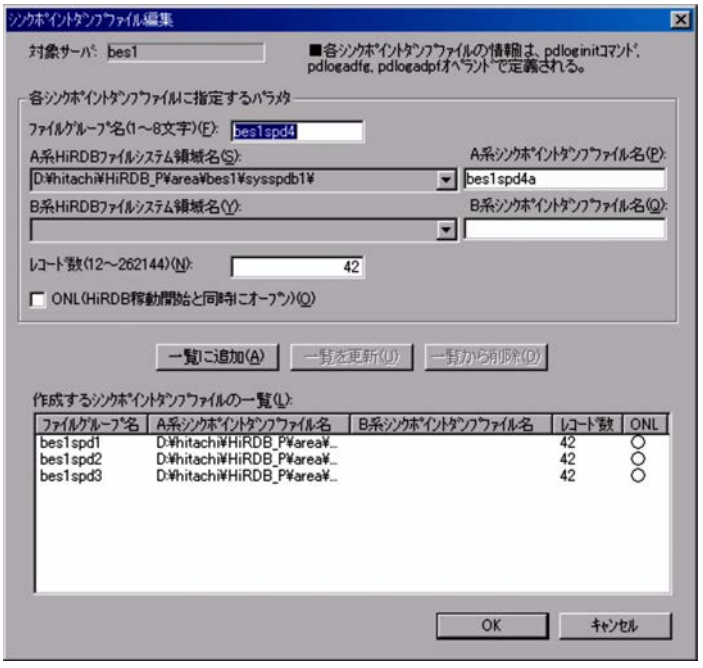
このダイアログボックスは，システム構成ツリービューで各サーバの定義を選択した状態で，メニューの［編集］－［編集］－［システムログファイル］又は［シンクポイントダンプファイル］を選択すると表示されます。

**●システムログファイル**




作成できるファイルグループは 200 個までのため，［作成するシステムログファイルの一覧］リストのファイルグループが 200 個になると，［一覧に追加］ボタンはクリックできなくなります。また，［作成するシステムログファイルの一覧］リストのファイルグループが 2 個（pd_log_rerun_swap=Y，又は pd_spd_assurance_count=2 の場合は 3 個）になると，［一覧から削除］ボタンはクリックできなくなります。

**●シンクポイントダンプファイル**




作成できるファイルグループは 60 個までのため，［作成するシンクポイントダンプファイルの一覧］リストのファイルグループが 60 個（［ONL（HiRDB 稼動開始と同時にオープン）］をチェックしている場合は 30 個）になると，［一覧に追加］ボタンはクリックできなくなります。また，［作成するシンクポイントダンプファイルの一覧］リストのファイルグループが 2 個になると，［一覧から削除］ボタンはクリックできなくなります。

システムログファイル又はシンクポイントダンプファイルを二重化（pd_log_dual 又は pd_spd_dual オペランドに Y を指定）する場合，B 系 HiRDB ファイルシステム領域名を指定してください。なお，二重化しているファイルと二重化していないファイルを［作成するシステムログファイルの一覧］リストに混在させることはできません。混在した状態で［OK］をクリックするとエラーになります。どちらかに統一してください。

このダイアログボックスで指定する項目について説明します。

| 項目名 | 説明 |
|---|---|
| **ファイルグループ名** | 作成するシステムログファイル又はシンクポイントダンプファイルのファイルグループ名を英数字 1 ～ 8 文字で指定します。初期値を次に示します。<br>システムログファイルの場合<br>[ **サーバ名** ]log[ **ファイルグループ数の通番** ]<br>ただし，8 文字を超える場合は次の名称とします。<br>[ **サーバ種別**※（**1 文字**）][ **サーバの通番** ]l[ **ファイルグループ数の通番** ]<br>シンクポイントダンプファイルの場合<br>[ **サーバ名** ]spd[ **ファイルグループ数の通番** ]<br>ただし，8 文字を超える場合は次の名称とします。<br>[ **サーバ種別**※（**1 文字**）][ **サーバの通番** ]d[ **ファイルグループ数の通番** ]<br>注※<br>d：ディクショナリサーバ<br>f：フロントエンドサーバ<br>b：バックエンドサーバ |
| A **系** HiRDB **ファイルシステム領域名** | 作成する A 系システムログファイル又は A 系シンクポイントダンプファイルを格納する HiRDB ファイルシステム領域名をドロップダウンリストで選択します。種別が SYS と SVR 以外の HiRDB ファイルシステム領域は選択できません。初期値を次に示します。<br>システムログファイルの場合<br>[ **ファイルの格納先** ]¥[ **サーバ名** ]¥syslog[ **サーバの種類** ][ **サーバ数の通番** ]¥<br>シンクポイントダンプファイルの場合<br>[ **ファイルの格納先** ]¥[ **サーバ名** ]¥sysspd[ **サーバの種類** ][ **サーバ数の通番** ]¥ |
| A **系システムログファイル名**又は A **系シンクポイントダンプファイル名** | 作成する A 系システムログファイル名又は A 系シンクポイントダンプファイルを英数字で指定します。初期値を次に示します。<br>システムログファイルの場合<br>[ **サーバ名** ]log[ **ファイルグループ数の通番** ]a<br>シンクポイントダンプファイルの場合<br>[ **サーバ名** ]spd[ **ファイルグループ数の通番** ]a |
| B **系** HiRDB **ファイルシステム領域名** | 作成する B 系システムログファイル又は B 系シンクポイントダンプファイルを格納する HiRDB ファイルシステム領域名をドロップダウンリストで選択します。種別が SYS と SVR 以外の HiRDB ファイルシステム領域は選択できません。システムログファイルを二重化する場合，障害発生に備えて，A 系と B 系のシステムログファイルは別々のハードディスクに作成することをお勧めします。A 系と異なるハードディスクの HiRDB ファイルシステム領域を選択してください。初期値は空です。 |
| B **系システムログファイル名**又は B **系シンクポイントダンプファイル名** | 作成する B 系システムログファイル名又は B 系シンクポイントダンプファイル名を英数字で指定します。初期値は空ですが，B 系 HiRDB ファイルシステム領域名が選択されると，「[ **サーバ名** ]log[ **ファイルグループ数の通番** ]b」又は「[ **サーバ名** ]spd[ **ファイルグループ数の通番** ]b」が表示されます。 |
| **レコード数** | システムログファイル又はシンクポイントダンプファイルのレコード数を指定します。 |
| ONL | このシステムログファイル又はシンクポイントダンプファイルのファイルグループを，HiRDB 開始と同時に使用できる状態（オープン状態）にする場合にチェックします。 |

［システムログファイル編集］又は［シンクポイントダンプファイル編集］ダイアログボックスで［OK］をクリックすると，［HiRDB セットアップツール―詳細定義（セットアップ）］ウィンドウへ戻ります。

### （6）必要に応じて定義の指定値を編集します

HiRDB システム定義の指定値を簡易セットアップツールの初期値から変更したい場合，HiRDB システム定義を編集します。システム構成ツリービューで作成する定義を選択します。オペランド情報リストビューに初期値が表示されるので，オペランドを追加又は削除したり，指定値を変更したりします。なお，HiRDB システム定義を編集する場合の注意事項については，「3.9　HiRDB システム定義を編集する場合の注意事項」を参照してください。簡易セットアップツールで作成できる定義については，「3.1.2(1) システム定義ファイル」を参照してください。

**注意事項**

簡易セットアップツールでは次の定義は作成できません。

- UAP 環境定義
- SQL 予約語定義
- 外部サーバ情報定義
- Hub 最適化情報定義

### (a) オペランド又はオプションフラグの追加

オペランド，又はオプションフラグを追加します。オプションフラグを追加できるのは，オペランド情報リストビューでコマンド形式のオペランドを選択している場合です。追加手順を次に示します。

**● オペランドの追加手順**

1. メニューの［編集］－［追加］－［オペランド］を選択します。
2. ［追加オペランド選択］ダイアログボックスのオペランドの一覧から，追加するオペランドを選択します。

3. ［OK］をクリックすると，［オペランド値編集］ダイアログボックスが表示されます。オペランド値の編集については，「3.6.2(6)(b) オペランド値の編集」を参照してください。

**● オプションフラグの追加手順**

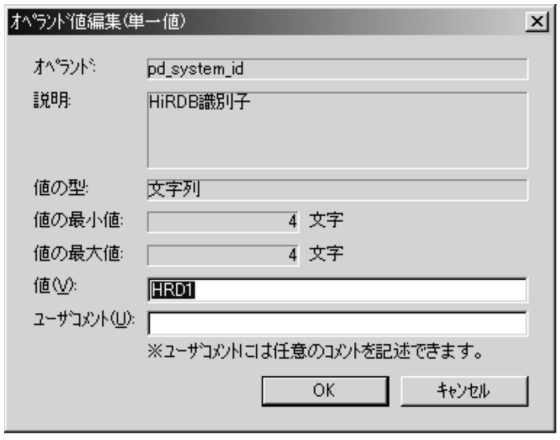
1. メニューの［編集］－［追加］－［オプションフラグ］を選択します。
2. ［追加オプションフラグ選択］ダイアログボックスのオプションフラグの一覧から，追加するオプションフラグを選択します。

3. ［OK］をクリックすると，［オペランド値編集］ダイアログボックスが表示されます。オペランド値の編集については，「3.6.2(6)(b) オペランド値の編集」を参照してください。

### (b) オペランド値の編集

オペランドの指定値を編集します。

1. 次のどちらかの操作をして，オペランドの形式に応じた［オペランド値編集］ダイアログボックスを表示させます。
   - メニューの［編集］－［編集］－［オペランド］を選択
   - ［追加オペランド選択］及び［追加オプションフラグ選択］ダイアログボックスでオペランドを選択して［OK］をクリック




なお，オペランドの形式によってタイトルバーの表示が異なります。「単一値」，「複数値」，「単一選択」，「複数選択」，及び「単一選択または任意」の 5 種類のダイアログボックスが表示されます。

2. オペランドの形式に応じてオペランドの指定値を編集します。
   指定値が一つの場合，オペランドの値を指定します。指定値を選択する場合，ドロップダウンリストになっているので，一つ又は複数選択します。
   指定値を一つ選択するか，又は任意の値を入力して編集する場合，ドロップダウンリスト中の選択肢から値を選択します。またテキスト部に任意の値を直接入力することもできます。
3. 必要に応じてユーザコメントを入力します。

ユーザコメントは任意です。コマンド形式オペランドのオプションフラグの場合は，そのオペランド全体に対するコメントになります。最大で半角 78 文字まで入力できます。

#### (c) オペランドの削除

削除するオペランドを，オペランド情報リストビューから選択して，メニューの［編集］－［削除］を選択します。

### (7) グローバルバッファを定義します

システム構成ツリービューで［システム共通定義］を選択した状態で，メニューの［編集］－［編集］－［グローバルバッファ］を選択すると，［グローバルバッファ編集］ダイアログボックスが表示されます。

簡易セットアップツールで定義するグローバルバッファを，［グローバルバッファの一覧］リストに［一覧に追加］で追加します。このダイアログボックス表示時は，初期値のグローバルバッファが表示されるので，リストで選択して，［一覧を更新］又は［一覧から削除］で編集したり，削除したりできます。また，作成するグローバルバッファを割り当てる RD エリアを［グローバルバッファを割り当てる RD エリアの指定］ダイアログボックスで指定します。

［グローバルバッファ編集］ダイアログボックスで［OK］をクリックすると，［HiRDB セットアップツール－詳細定義（セットアップ）］ウィンドウへ戻ります。

**●グローバルバッファの定義**

［グローバルバッファ編集］ダイアログボックスで指定する項目について説明します。

| 項目名 | 説明 |
|---|---|
| **バッファ名** | グローバルバッファの名称を 1 ～ 16 文字で指定します。 |
| **バッファ面数** | グローバルバッファの面数を指定します。 |
| **残りの全ての RD エリアにグローバルバッファを割り当てる** | pdbuffer オペランドの -r オプションで指定していない全 RD エリアにグローバルバッファを割り当てる場合にチェックします。なお，この項目をチェックしていると，［グローバルバッファを割り当てる RD エリアの指定］はクリックできません。 |

**●作成したグローバルバッファを割り当てる RD エリアの指定**

［グローバルバッファ編集］ダイアログボックスの［グローバルバッファを割り当てる RD エリアの指定］をクリックすると，［グローバルバッファを割り当てる RD エリアの指定］ダイアログボックスが表示されます。

グローバルバッファを割り当てる RD エリアを［残りの RD エリア］（グローバルバッファが割り当てられていない RD エリアの一覧）から［割り当てる RD エリア］に［追加］します。又は，［割り当てる RD エリア］から［削除］します。［OK］をクリックすると，［グローバルバッファ編集］ダイアログボックスに戻ります。

## (8) HiRDB システム定義のチェックをします

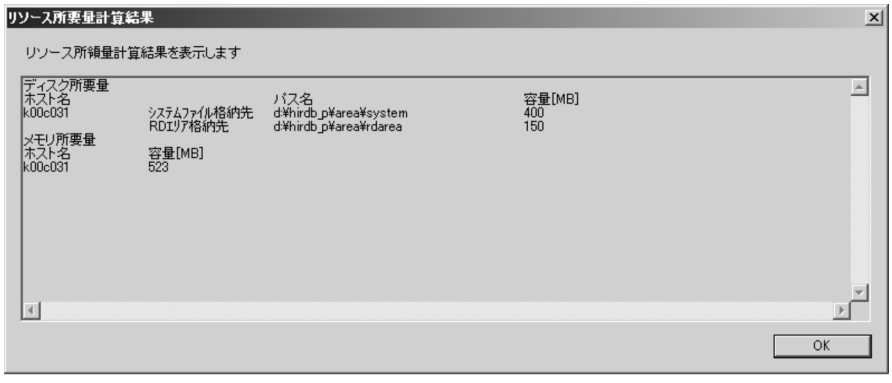
HiRDB システム定義の値を変更した場合，メニューの［ファイル］－［システム定義のチェック］を選択して，HiRDB システム定義のチェックをする必要があります。チェックをしないと指定値の変更は有効になりません。チェック結果は［システム定義のチェック結果］ダイアログボックスに表示されます。

HiRDB システム定義のチェックでエラーがあると，セットアップはできません。エラーの原因を取り除いてから再度 HiRDB システム定義のチェックをしてください。

## (9) リソース所要量を確認します

簡易セットアップツールで設定した HiRDB システム定義で必要となるメモリ所要量とディスク容量を計算できます。セットアップを開始する前にメモリ所要量とディスク容量が十分かどうか確認してください。

メニューの［リソース］－［リソース所要量の計算］を選択すると，［リソース所要量計算結果］ダイアログボックスに計算結果が表示されます。計算はメガバイト単位で行い，端数は繰り上げます。




ディスク容量は，「システムファイル格納先」と「RD エリア格納先」が同じ場合は「セットアップディレクトリ」として表示し，異なる場合は別々に表示します。また，「システムファイル格納先」と「RD エリア格納先」以外に作成するように指定した場合，「その他」として，ドライブごとの合計値を表示します。

［リソース所要量結果］ダイアログボックスを表示したまま，ほかのシステム定義を設定するダイアログボックスで指定を変更できます。オペランドを追加又は値を編集して，メニューの［リソース］－［リソース所要量の計算］を選択すると計算結果が更新されます。［HiRDB セットアップツール－詳細定義］ウィンドウを閉じると，［リソース所要量結果］ダイアログボックスも閉じます。

計算対象とするリソースを次の表に示します。

**表 3-3　リソース所要量の計算対象となるリソース**

| リソース | |
|---|---|
| メモリ所要量 | ユニットコントローラが使用する共用メモリ |
| | 各サーバが使用する共用メモリ |
| | グローバルバッファが使用する共用メモリ |
| RD エリアの容量 | ユーザ用 RD エリアの容量 |
| | データディクショナリ用 RD エリアの容量 |
| | マスタディレクトリ用 RD エリアの容量 |
| | データディレクトリ用 RD エリアの容量 |
| | データディクショナリ用 RD エリアの容量 |
| | データディクショナリ LOB 用 RD エリアの容量 |
| | ユーザ LOB 用 RD エリアの容量 |
| | リスト用 RD エリアの容量 |
| システムファイルの容量 | システムログファイルの容量 |
| | シンクポイントダンプファイルの容量 |
| | ステータスファイルの容量 |
| 作業表用ファイルの容量 | 作業表用ファイルの容量 |
| ユティリティ実行時の容量 | ユティリティ用の HiRDB ファイルシステムの容量 |

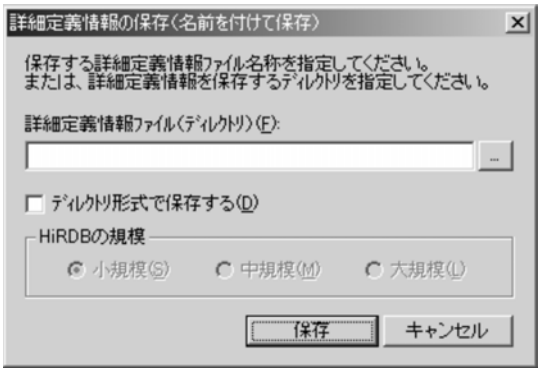
メモリ所要量は，指定された定義オペランドを基に見積もります。また，ディスク所要量は，［HiRDB ファイルシステム領域編集］ダイアログボックスで作成するように指定された HiRDB ファイルシステム領域について，指定値を基に見積もります。

なお，ここで表示されるのは，HiRDB 構築時のディスク所要量です。HiRDB 開始後の運用によっては，規模が拡大する場合があります。その場合の詳細については，「3.6.2(4)(a)　HiRDB ファイルシステム領域を編集します」の注を参照してください。

## (10) 詳細定義情報を保存します

**名前を付けて保存する場合**

設定した詳細定義情報をローカルディスク上に名前を付けて保存する場合は，メニューの［ファイル］－［詳細定義情報の保存］を選択します。
［詳細定義情報の保存］ダイアログボックスが表示されます。




詳細定義情報ファイル（ディレクトリ）名を指定します。
このダイアログボックスで指定する項目について説明します。

| 項目名 | 説明 |
|---|---|
| **詳細定義情報ファイル（ディレクトリ）** | 詳細定義情報ファイル名又はディレクトリ名をフルパスで指定します。ファイル形式で保存する場合，拡張子は dat としてください。 |
| **ディレクトリ形式で保存する** | 詳細定義情報をディレクトリ形式で保存する場合にチェックします。 |
| **HiRDB の規模** | ［ディレクトリ形式で保存する］をチェックしている場合に選択できます。<br>ディレクトリ構造形式では，異なる規模の詳細定義情報を同じディレクトリ下に保存できます。ただし，サーバ種別が異なる場合（HiRDB/ パラレルサーバと HiRDB/ シングルサーバなど）は，同じディレクトリ下には保存できません。 |

［保存］をクリックすると，確認メッセージを表示した後に詳細定義情報を保存し，［HiRDB セットアップツール－詳細定義（セットアップ)］ウィンドウに戻ります。

**上書き保存する場合**

設定した詳細定義情報をローカルディスク上に上書き保存する場合は，メニューの［ファイル］－［詳細定義情報の上書き保存］を選択します。確認メッセージを表示した後に詳細定義情報を保存し，［HiRDB セットアップツール－詳細定義（セットアップ)］ウィンドウに戻ります。

## (11) セットアップを開始します

メニューの［ファイル］－［終了］を選択すると，［HiRDB セットアップツール－カスタム］ウィンドウに戻ります※。［セットアップ開始］をクリックすると，セットアップが開始されます。

注※

ウィザードセットアップで環境設定をした後に詳細定義を行った場合は，［HiRDB セットアップツール－ウィザードセットアップ［確認]］ウィンドウに戻ります。

# 3.7 詳細定義情報の読み込み

ローカルディスク上に保存した詳細定義情報を読み込みます。

［HiRDB セットアップツール－開始］ウィンドウで［詳細定義情報の読み込み］をクリックすると，［詳細定義情報の読み込み］ダイアログボックスが表示されます。そこで詳細定義情報を読み込むと，［HiRDB セットアップツール－詳細定義（セットアップ）］ウィンドウが開きます。

詳細定義情報の読み込みについては，「3.6.2(2)　詳細定義情報を読み込みます」を参照してください。

**注意事項**

- ［HiRDB セットアップツール－開始］ウィンドウから［詳細定義情報の読み込み］を選択して詳細定義情報を読み込む場合，［システムファイル格納先ディレクトリ］と［RD エリア格納先ディレクトリ］として，［セットアップディレクトリ］が使用されます。
- バージョン 08-02 の簡易セットアップツールで保存した詳細定義情報を，バージョン 08-03 以降の簡易セットアップツールで読み込むことができます。

# 3.8　定義の更新

作成済みの HiRDB システム定義を更新します。既存の HiRDB システムがない場合は選択できません。なお，簡易セットアップツールを使って設定していない環境の場合，HiRDB システム定義ファイルの記述方法によっては，簡易セットアップツールが対応できないでエラーとなるときがあります。このようなときは，手動で HiRDB システム定義を更新してください。

## (1) 定義を更新する前に

定義を更新する場合，HiRDB を停止し，アンロード待ち状態のシステムログファイルをアンロードしておく必要があります。また，次のオペランドの変更時は pdloginit コマンドでシステムログファイルを初期化する必要があります。

- pd_log_dual
- pdstart

## (2) 定義の更新手順

定義の更新手順を次に示します。

1. ［HiRDB セットアップツール－開始］ウィンドウで［定義更新］をクリックします。
   ［HiRDB セットアップツール－詳細定義（定義更新）］ウィンドウが表示されます。
2. システム構成ツリービューで作成する定義を選択して，オペランドを追加又は削除したり，指定値を変更したりします。なお，HiRDB システム定義を編集する場合の注意事項については，「3.9　HiRDB システム定義を編集する場合の注意事項」を参照してください。
3. メニューの［ファイル］－［システム定義のチェック］を選択して，HiRDB システム定義のチェックをします。
   HiRDB システム定義のチェックでエラーがあると，定義の更新はできません。エラーの原因を取り除いてから再度 HiRDB システム定義のチェックをしてください。
4. メニューの［ファイル］－［システム定義の更新］を選択すると，HiRDB システム定義が更新されます。
5. メニューの［ファイル］－［終了］を選択すると，定義の更新を終了します。

**注意事項**

- 簡易セットアップツールはファイルの文字コード種別を SJIS として扱います。システム定義ファイルを SJIS 以外で保存した場合，［システム定義の更新］でオペランドのコメント欄の文字コード種別依存文字が GUI 画面上で文字化けすることがありますが，［システム定義の更新］に影響ありません（［システム定義のチェック］は正常終了します）。
- 文字コード種別を UTF-8 でファイル保存した場合，ファイル先頭に UTF-8 を示す BOM（Byte Order Mark：0xEFBBBF）が付加されることがあります。HiRDB を文字コード種別 UTF-8 でインストールしている場合だけ，簡易セットアップツールはシステム定義の更新時にユーザが作成したシステム定義ファイルの BOM を読みとばして，システム定義の更新実行後のファイルからは BOM を削除します。UTF-8 以外の文字コードでセットアップしている場合は，BOM はデータの一部として読み込みます。そのため，［HiRDB セットアップツール－詳細定義（セットアップ）］ウィンドウのオペランド一覧に不正なオペランドが表示されることがあり，［システム定義のチェック］がエラーとなってシステム定義の更新を実行することはできません。

# 3.9 HiRDB システム定義を編集する場合の注意事項

HiRDB システム定義を編集する場合の注意事項を次に示します。

## (1) ファイル名の長さ

ファイル名はフルパスで 167 バイト以下にしてください。特に HiRDB/ パラレルサーバでは，システムファイル及び RD エリアを次のディレクトリ下に作成するため，注意が必要です。

- ユーザが指定したセットアップディレクトリ ¥fes1¥
- ユーザが指定したセットアップディレクトリ ¥dic1¥
- ユーザが指定したセットアップディレクトリ ¥bes1¥
- ユーザが指定したセットアップディレクトリ ¥bes2¥

ただし，ステータスファイルの編集，システムログファイルの編集，シンクポイントダンプファイルの編集のダイアログでファイルの追加又は編集を行う場合は，64 バイト以下にしてください。

## (2) pd_system_id オペランドの変更

システム共通定義（pdsys）の pd_system_id オペランドは，セットアップ時だけ変更できます。定義更新時に編集しようとするとエラーになります。

## (3) pd_unit_id オペランドの変更

ユニット制御情報定義（pdutsys）の pd_unit_id オペランド（ユニット識別子）を変更すると，ユニット識別子を指定しているシステム共通定義の次のオペランドの値も自動的に更新されます。

- pdunit -u（ユニット識別子）
- pdstart -u（ユニット識別子）

## (4) マシン情報の変更

［マシン情報編集］ダイアログボックスでホスト名や HiRDB 運用ディレクトリを変更すると，それらの値を指定するシステム共通定義の次のオペランドの値も自動的に更新されます。

- pdunit -x（ホスト名）
- pdunit -d（HiRDB 運用ディレクトリ）
- pdunit -c（ホスト名）
- pdstart -x（ホスト名）
- pdstart -m（ホスト名）
- pdstart -n（ホスト名）

## (5) サーバ名の変更

サーバ名を変更すると，サーバ名を指定する pdstart オペランドの -s オプションの値も自動的に更新されます。

## (6) ユニットの追加，又は削除

ユニットを追加，又は削除すると，pdunit オペランドの値も自動的に追加，又は削除されます。

## (7) サーバの追加，又は削除

サーバを追加，又は削除すると，pdstart オペランドの値も自動的に追加，又は削除されます。

### (8) マシン情報，ユニット，又はサーバのコピーと貼り付け

マシン情報，ユニット，又はサーバをコピーして貼り付けた場合，次の内容はコピー元の定義内容を引き継ぎます。

- HiRDB ファイルシステム領域のパス
- RD エリアの情報（RD エリア名，サーバ名，及び RD エリアに割り当てた HiRDB ファイル）
- 各サーバの情報（ステータスファイル，システムログファイル，及びシンクポイントダンプファイル）
- 各サーバのサーバ定義中の各オペランド値

### (9) サーバ名とマシン情報の HiRDB 運用ディレクトリの変更

サーバ名とマシン情報の HiRDB 運用ディレクトリを変更した場合も，(8) に示す情報は自動的に変更後の値に更新されます。

# 3.10 HiRDB/ パラレルサーバの環境設定をする場合

簡易セットアップツールの初期値では，サーバマシンが一つの構成のため，HiRDB/ パラレルサーバの場合はサーバマシンを追加することから始めます。ここでは，バックエンドサーバ（BES）を追加する例を示しながら HiRDB/ パラレルサーバの環境設定手順を説明します。

## （1） サーバマシンを追加します

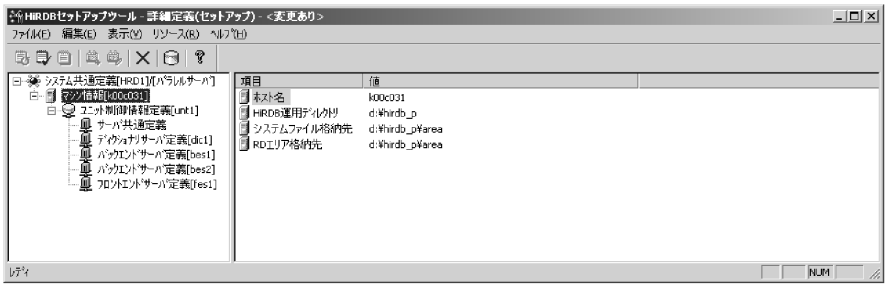
［HiRDB セットアップツール－詳細定義（セットアップ）］ウィンドウで，次のどちらかの方法でサーバマシンを追加します。

1. 新規追加
   システム構成ツリービューで［システム共通定義］を選択した状態で，メニューの［編集］－［追加］－［マシン］を選択します。
   ［マシン情報編集］ダイアログボックスが表示されるので，追加するマシンの情報を設定します。［マシン情報編集］ダイアログボックスについては，「3.6.2(3)(a) マシン情報の編集」を参照してください。
2. 既存のマシン情報をコピー
   システム構成ツリービューで既存の［マシン情報］を選択した状態で，メニューの［編集］－［コピー］を選択します。システム構成ツリービューで［システム共通定義］を選択した状態で，メニューの［編集］－［貼り付け］を選択します。
   コピーしたサーバマシンと同じ内容のサーバマシン情報が追加されます。不要な情報は削除します。

1. の場合は HiRDB ファイルシステム領域や RD エリアの情報などが設定されません。2. の場合は既存の情報がそのままコピーされ，既存の情報を基に編集できるため，2. をお勧めします。

サーバマシンを追加した［HiRDB セットアップツール－詳細定義（セットアップ）］ウィンドウを次の図に示します。

**図 3-10 サーバマシンの追加例**




## （2） 追加した BES の情報を編集します

［マシン情報編集］ダイアログボックスで，追加した BES のホスト名，HiRDB 運用ディレクトリ，システムファイル及び RD エリア格納先の情報を編集します。

## （3） 追加した BES の HiRDB ファイルシステム領域を編集します

システム構成ツリービューで，追加した BES の［ユニット制御情報定義］を選択した状態で，メニューの［編集］－［編集］－［HiRDB ファイルシステム領域］を選択して，［HiRDB ファイルシステム領域編集］ダイアログボックスを表示し，各 HiRDB ファイルシステム領域の情報を編集します。ここでは，次に示す値を指定します。

**● 追加した BES の HiRDB ファイルシステム領域の定義（ユニット制御情報定義 unt2 の定義）**

| ファイル名※ | 領域サイズ | 最大ファイル数 | 使用目的 | 増分回数 | 初期化 |
|---|---|---|---|---|---|
| %PDDIR%¥area¥unt2¥sysstsut | 5 | 10 | SYS | 1 | ○ |
| %PDDIR%¥area¥bes3¥rdworkb3 | 25 | 10 | WORK | 255 | ○ |
| %PDDIR%¥area¥bes3¥rddatb3 | 20 | 10 | DB | 1 | ○ |
| %PDDIR%¥area¥bes3¥rdlobb3 | 10 | 10 | DB | 1 | ○ |
| %PDDIR%¥area¥bes3¥sysstsb3 | 30 | 10 | SYS | 1 | ○ |
| %PDDIR%¥area¥bes3¥sysspdb3 | 5 | 10 | SYS | 1 | ○ |
| %PDDIR%¥area¥bes3¥syslogb3 | 70 | 10 | SYS | 1 | ○ |

（凡例）
○：［領域を初期化する（-i オプション）］チェックボックスをチェックしています。

注※　%PDDIR% は，追加したサーバマシンの HiRDB 運用ディレクトリです。

なお，既存のサーバマシンをコピーしてサーバマシン情報を追加した場合，コピー元のサーバマシンの情報が設定されているため，上記以外の情報をすべて削除してください。

## (4) 追加した BES の RD エリアを編集します

システム構成ツリービューで，追加した BES の［ユニット制御情報定義］を選択した状態で，メニューの［編集］－［編集］－［RD エリア］を選択して，［RD エリア編集］ダイアログボックスを表示し，各 RD エリアの情報を編集します。ここでは，次に示す値を指定します。

**● 追加した BES の RD エリアの定義（ユニット制御情報定義 unt2 の定義）**

- 各 RD エリアの情報

| RD エリア名 | RD エリアの種類 | サーバ名 | ページ長 | セグメントサイズ |
|---|---|---|---|---|
| bes3data | user used by PUBLIC | bes3 | 4,096 | 10 |
| bes3indx | user used by PUBLIC | bes3 | 4,096 | 10 |
| bes3lob | LOB used by PUBLIC | bes3 | 8,192 | 1 |

- 各 RD エリアに割り当てる HiRDB ファイルの情報

| RD エリア名 | HiRDB ファイル名※ | セグメント数 |
|---|---|---|
| bes3data | %PDDIR%¥area¥bes3¥rddatb3¥bes3data | 300 |
| bes3indx | %PDDIR%¥area¥bes3¥rddatb3¥bes3indx | 100 |
| bes3lob | %PDDIR%¥area¥bes3¥rddatb3¥bes3lob | 400 |

注※　%PDDIR% は，追加したサーバマシンの HiRDB 運用ディレクトリです。

なお，既存のサーバマシンをコピーしてサーバマシン情報を追加した場合，コピー元のサーバマシンの情報が設定されているため，上記以外の情報をすべて削除してください。

## (5) 追加した BES のユニットステータスファイルを編集します

システム構成ツリービューで，追加した BES の［ユニット制御情報定義］を選択した状態で，メニューの［編集］－［編集］－［ステータスファイル］を選択して，［ステータスファイル編集］ダイアログボッ

クスを表示し，各ステータスファイルの情報を編集します。ここでは，次に示す値を指定します。

**● 追加した BES のユニットステータスファイルの定義（ユニット制御情報定義 unt2 の定義）**

| 論理ファイル名 | A 系ステータスファイル名※ | B 系ステータスファイル名※ | レコード長 | レコード数 |
|---|---|---|---|---|
| ut2sts1 | %PDDIR%¥area¥unt2¥sysstsut¥ut2sts1a | %PDDIR%¥area¥unt2¥sysstsut¥ut2sts1b | 4,096 | 32 |
| ut2sts2 | %PDDIR%¥area¥unt2¥sysstsut¥ut2sts2a | %PDDIR%¥area¥unt2¥sysstsut¥ut2sts2b | 4,096 | 32 |
| ut2sts3 | %PDDIR%¥area¥unt2¥sysstsut¥ut2sts3a | %PDDIR%¥area¥unt2¥sysstsut¥ut2sts3b | 4,096 | 32 |

注※　%PDDIR% は，追加したサーバマシンの HiRDB 運用ディレクトリです。

## (6) 追加した BES のステータスファイルを編集します

システム構成ツリービューで，追加した BES の定義を選択した状態で，メニューの［編集］－［編集］－［ステータスファイル］を選択して，［ステータスファイル編集］ダイアログボックスを表示し，各ステータスファイルの情報を編集します。ここでは，次に示す値を指定します。

**● 追加した BES のサーバステータスファイルの定義（バックエンドサーバ定義 bes3 の定義）**

| 論理ファイル名 | A 系ステータスファイル名※ | B 系ステータスファイル名※ | レコード長 | レコード数 |
|---|---|---|---|---|
| bes3sts1 | %PDDIR%¥area¥bes3¥sysstsb3¥bes3sts1a | %PDDIR%¥area¥bes3¥sysstsb3¥bes3sts1b | 4,096 | 800 |
| bes3sts2 | %PDDIR%¥area¥bes3¥sysstsb3¥bes3sts2a | %PDDIR%¥area¥bes3¥sysstsb3¥bes3sts2b | 4,096 | 800 |
| bes3sts3 | %PDDIR%¥area¥bes3¥sysstsb3¥bes3sts3a | %PDDIR%¥area¥bes3¥sysstsb3¥bes3sts3b | 4,096 | 800 |

注※　%PDDIR% は，追加したサーバマシンの HiRDB 運用ディレクトリです。

## (7) 追加した BES のシステムログファイルを編集します

システム構成ツリービューで，追加した BES の定義を選択した状態で，メニューの［編集］－［編集］－［システムログファイル］を選択して，［システムログファイル編集］ダイアログボックスを表示し，各システムログファイルの情報を編集します。ここでは，次に示す値を指定します。

**● 追加した BES のシステムログファイルの定義（バックエンドサーバ定義 bes3 の定義）**

| ファイルグループ名 | A 系システムログファイル名※ | B 系システムログファイル名 | レコード数 | ONL |
|---|---|---|---|---|
| bes3log1 | %PDDIR%¥area¥bes3¥syslogb3¥bes3log1 | 指定なし | 16,000 | ○ |
| bes3log2 | %PDDIR%¥area¥bes3¥syslogb3¥bes3log2 | 指定なし | 16,000 | ○ |
| bes3log3 | %PDDIR%¥area¥bes3¥syslogb3¥bes3log3 | 指定なし | 16,000 | ○ |
| bes3log4 | %PDDIR%¥area¥bes3¥syslogb3¥bes3log4 | 指定なし | 16,000 | ○ |

（凡例）

○：［ONL（HiRDB 稼動開始と同時にオープン）］チェックボックスをチェックしています。

注※　%PDDIR% は，追加したサーバマシンの HiRDB 運用ディレクトリです。

### (8) 追加した BES のシンクポイントダンプファイルを編集します

システム構成ツリービューで，追加した BES 定義を選択した状態で，メニューの［編集］－［編集］－［シンクポイントダンプファイル］を選択して，［シンクポイントダンプファイル編集］ダイアログボックスを表示し，各シンクポイントダンプファイルの情報を編集します。ここでは，次に示す値を指定します。

**● 追加した BES のシンクポイントダンプファイルの定義（バックエンドサーバ定義 bes3 の定義）**

| ファイルグループ名 | シンクポイントダンプファイル名※ | レコード数 | ONL |
|---|---|---|---|
| bes3spd1 | %PDDIR%¥area¥bes3¥sysspdb3¥bes3spd1 | 42 | ○ |
| bes3spd2 | %PDDIR%¥area¥bes3¥sysspdb3¥bes3spd2 | 42 | ○ |
| bes3spd3 | %PDDIR%¥area¥bes3¥sysspdb3¥bes3spd3 | 42 | ○ |

（凡例）
　○：［ONL（HiRDB 稼動開始と同時にオープン）］チェックボックスをチェックしています。

注※　%PDDIR% は，追加したサーバマシンの HiRDB 運用ディレクトリです。

### (9) pd_log_auto_unload_path オペランドを編集します

追加した BES の pd_log_auto_unload_path オペランドを編集し，自動ログアンロード機能を使用するようにします。アンロードログファイルの出力先ディレクトリを指定します。ここでは，バックエンドサーバ定義［bes3］に「%PDDIR%¥area¥bes3¥unloadlog¥」を指定します。%PDDIR% は，追加した HiRDB サーバの HiRDB 運用ディレクトリです。

### (10) pdwork -v オペランドを編集します

追加した BES の pdwork -v オペランドを編集し，作業表用ファイル用の HiRDB ファイルシステム領域の名称を指定します。ここでは，バックエンドサーバ定義［bes3］に「%PDDIR%¥area¥bes3¥rdworkb3」を指定します。%PDDIR% は，追加した HiRDB サーバの HiRDB 運用ディレクトリです。

### (11) 追加した BES の RD エリアにグローバルバッファを割り当てます

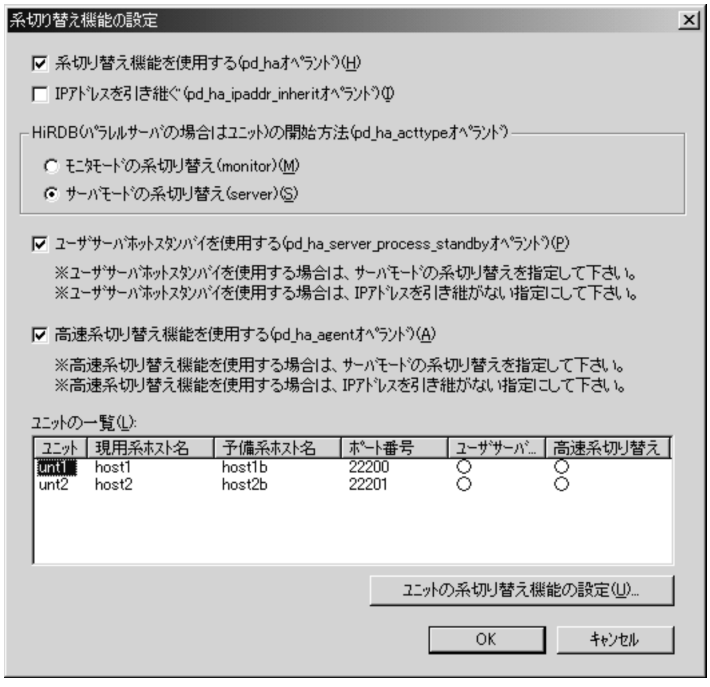
［グローバルバッファ編集］ダイアログボックスを表示し，追加した BES の RD エリアにグローバルバッファを割り当てます。ここでは，特に編集しないで，［残りの全ての RD エリアにグローバルバッファを割り当てる］チェックボックスをチェックします。［残りの全ての RD エリアにグローバルバッファを割り当てる］をチェックする場合，バッファ面数は十分確保してください。［バッファ面数（4 ～ 460000）］を指定しないときは 10,000 が仮定されます。

### (12) HiRDB システム定義をチェックし，セットアップを開始します

設定した HiRDB システム定義をチェックして，問題がなければセットアップを開始します。

# 3.11 系切り替え機能の設定

系切り替え機能を使用する場合，システム構成ツリービューで［システム共通定義］を選択した状態で，メニューの［編集］－［編集］－［系切り替え機能の設定］を選択します。［系切り替え機能の設定］ダイアログボックスが表示されるので，HiRDB システム定義の系切り替え機能に関するオペランドを設定します。




簡易セットアップツールでは次の作業ができます。

- HiRDB システム定義の系切り替え機能に関するオペランドの設定
- 共有ディスクへの HiRDB ファイルシステム領域の作成
- HiRDB システム定義ファイルの予備系への配布

簡易セットアップツールでの系切り替え機能の設定可否を次に示します。

| 系切り換え機能 | | 設定可否 |
|---|---|---|
| スタンバイ型系切り替え | ユーザサーバホットスタンバイ | ○ |
| | 高速系切り替え機能 | ○ |
| スタンバイレス型系切り替え | 1：1 スタンバイレス型系切り替え機能 | × |
| | 影響分散スタンバイレス型系切り替え機能 | × |

（凡例）
○：設定できます。
×：設定できません。

系切り替え機能の詳細については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してくだ

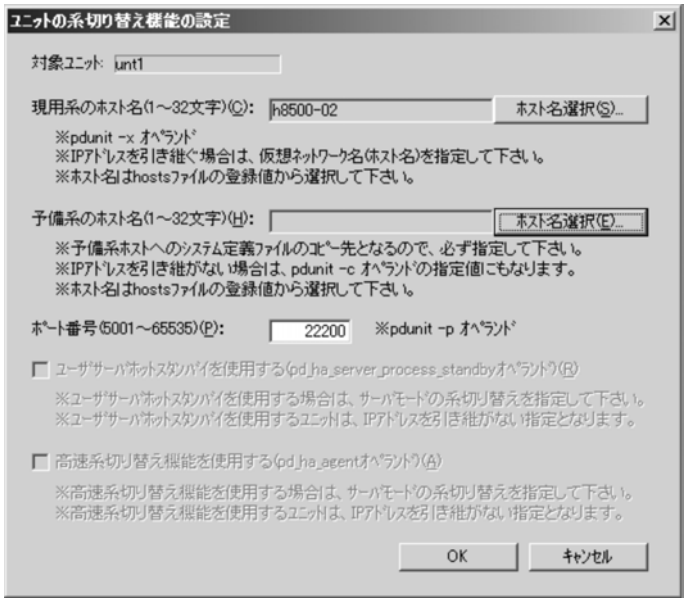
さい。

### (1) ［系切り替え機能の設定］ダイアログボックスで指定する項目

［系切り替え機能の設定］ダイアログボックスで指定する項目について説明します。

| 項目名 | 説明 | 注意事項 |
|---|---|---|
| **系切り替え機能を使用する** | 系切り替え機能を使用するかどうかを指定します。系切り替え機能を使用する場合はチェックします。また，系切り替え機能を使用する場合は，各ユニット制御情報定義（pdutsys）の pd_hostname オペランドに現用系の標準ホスト名（マシン情報のホスト名）が自動的に設定されます。 | 定義更新時は選択できません。 |
| IP **アドレスを引き継ぐ** | 系切り替え時に IP アドレスを引き継ぐかどうかを指定します。IP アドレスを引き継ぐ場合はチェックします。 | この項目をチェックすると，次のチェックボックスは選択できません。<br>• ［ユーザサーバホットスタンバイを使用する］<br>• ［高速系切り替え機能を使用する］ |
| HiRDB**（パラレルサーバの場合はユニット）の開始方法** | HiRDB（HiRDB/ パラレルサーバの場合はユニット）の開始方法を指定します。ここで指定した方法で全ユニットが開始します。 | ［モニタモードの系切り替え］を指定した場合，次のチェックボックスは選択できません。<br>• ［ユーザサーバホットスタンバイを使用する］<br>• ［高速系切り替え機能を使用する］ |
| **ユーザサーバホットスタンバイを使用する** | ユーザサーバホットスタンバイを使用するかどうかを指定します。使用する場合はチェックします。チェックすると，全ユニットに対してユーザサーバホットスタンバイを使用します。 | 次のように指定していない場合，この項目は選択できません。<br>• IP アドレスを引き継がない<br>• サーバモードの系切り替え |
| **高速系切り替え機能を使用する** | 高速系切り替え機能を使用するかどうかを指定します。使用する場合はチェックします。チェックすると，全ユニットに対して高速系切り替え機能を使用します。 | 次のように指定していない場合，この項目は選択できません。<br>• IP アドレスを引き継がない<br>• サーバモードの系切り替え |

［系切り替え機能の設定］ダイアログボックスの［ユニット一覧］には，各ユニットの系切り替えに関する情報が表示されます。［ユニット一覧］でユニットを選択して，［ユニットの系切り替え機能の設定］をクリックすると，［ユニットの系切り替え機能の設定］ダイアログボックスが表示されます。［ユニット一覧］に選択項目がない場合は選択できません。




このダイアログボックスで，現用系のホスト名，予備系のホスト名，ポート番号などを指定します。指定する項目について説明します。

| 項目名 | 説明 | 注意事項 |
|---|---|---|
| **現用系のホスト名** | 現用系のホスト名を 1 ～ 32 文字で指定します。IP アドレスを引き継ぐ場合は仮想ネットワーク名（現用系，予備系の両方で共通のホスト名）を指定します。［ホスト名選択］をクリックすると，［hosts ファイルからホスト名を選択］ダイアログボックスが表示され，ホスト名を選択できます。 | － |
| **予備系のホスト名** | 予備系のホスト名を 1 ～ 32 文字で指定します。予備系ホストへのシステム定義ファイルのコピー先となるので必ず指定してください。［ホスト名選択］をクリックすると，［hosts ファイルからホスト名を選択］ダイアログボックスが表示され，ホスト名を選択できます。 | － |
| **ポート番号** | 各ユニットのポート番号を指定します。 | HiRDB/ シングルサーバの場合は指定できません。 |
| **ユーザサーバホットスタンバイを使用する** | ユーザサーバホットスタンバイを使用するかどうかを指定します。使用する場合はチェックします。ユーザサーバホットスタンバイを使用する場合，そのユニットは IP アドレスを引き継がない指定になります。ユーザサーバホットスタンバイを使用しない場合，IP アドレスを引き継ぐかどうかは，［系切り替え機能の設定］ダイアログボックスの［IP アドレスを引き継ぐ］の指定に依存します。 | モニタモードの場合は選択できません。 |

| 項目名 | 説明 | 注意事項 |
|---|---|---|
| **高速系切り替え機能を使用する** | 高速系切り替え機能を使用するかどうかを指定します。使用する場合はチェックします。高速系切り替え機能を使用する場合，そのユニットは IP アドレスを引き継がない指定となります。高速切り替え機能を使用しない場合，IP アドレスを引き継ぐかどうかは，［系切り替え機能の設定］ダイアログボックスの［IP アドレスを引き継ぐ］の指定に依存します。 | モニタモードの場合，及び HiRDB が 06-02-/E より前のバージョンの場合は選択できません。 |

（凡例）－：該当しません。

## (2) 系切り替え機能を使用する場合の注意事項

簡易セットアップツールで系切り替えの設定をする場合の注意事項を次に示します。

- 簡易セットアップツールは現用系マシンで実行してください。
- 共有ディスクに HiRDB ファイルシステム領域を作成する場合，［HiRDB ファイルシステム領域編集］ダイアログボックスで，作成先のパスを共有ディスク上のパスに変更してください。なお，系切り替え機能使用時は次の制限があるので，確認してください。
  - pdfmkfs コマンドの -k オプションには SVR を指定しない
  - pdfmkfs コマンドの -i オプションは指定する
- 簡易セットアップツールは，セットアップ実行時及び HiRDB システム定義更新時に，自動的に予備系へシステム定義ファイルを配布します。このため，次のようにしてください。
  - 現用系及び予備系で HiRDB のサービス（HiRDB/SingleServer 又は HiRDB/ParallelServer）を起動しておく
  - サービスのログオンをユーザアカウントで起動する
  - HiRDB/ シングルサーバで 06-02-/E より前のバージョンの場合，コマンドプロンプトから pdntenv -ro on を実行し，リモート系コマンドを使用できる状態（予備系への配布ができる状態）にする
- 系切り替え機能を使用する場合，セットアップが完了しても HiRDB を開始しません。手動で開始してください。このとき，HiRDB は pdstart コマンド実行後に初期設定終了待ち状態になるため，この状態でデータベース初期設定ユティリティ（pdinit）を実行する必要があります。データベース初期設定ユティリティに指定する制御文ファイル名には簡易セットアップツールが作成したファイル（%PDCONFPATH%¥mkinit）を指定します。データベース初期設定ユティリティについては，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。
  また，系切り替え機能を使用する場合，サンプルデータベースは自動的に作成されません。サンプルデータベースが必要なときは HiRDB 開始後にサンプルデータベース作成用のバッチファイルを実行してください。サンプルデータベース作成用のバッチファイルについては，「3.1.2(3) サンプルデータベース関連ファイル」を参照してください。
- 系切り替え機能を使用する場合，システム共通定義（pdsys）の pd_mode_conf オペランドが AUTO のときは，自動的に MANUAL2 に変更されるので注意してください。
- HiRDB ファイルシステム領域の作成先となる共有ディスク（物理ディスク）は，クラスタアドミニストレータでオンライン状態にしておいてください。

# 4 コマンドによる環境設定

この章では，コマンドを使用して HiRDB の環境を設定する方法について説明します。


4.1 コマンドによる環境設定の概要

4.2 HiRDB システム定義の作成

4.3 HiRDB ファイルシステム領域の作成

4.4 システムファイルの作成

4.5 システム用 RD エリアの作成

4.6 HiRDB の初期開始

4.7 ユーザ用 RD エリアの作成

4.8 ユーザ LOB 用 RD エリアの作成

4.9 データディクショナリ LOB 用 RD エリアの作成

4.10 リスト用 RD エリアの作成

# 4.1 コマンドによる環境設定の概要

## (1) 環境設定の前に設計する項目

HiRDB の環境設定をする前に，システムの構成を設計してください。次に示す項目について設計します。

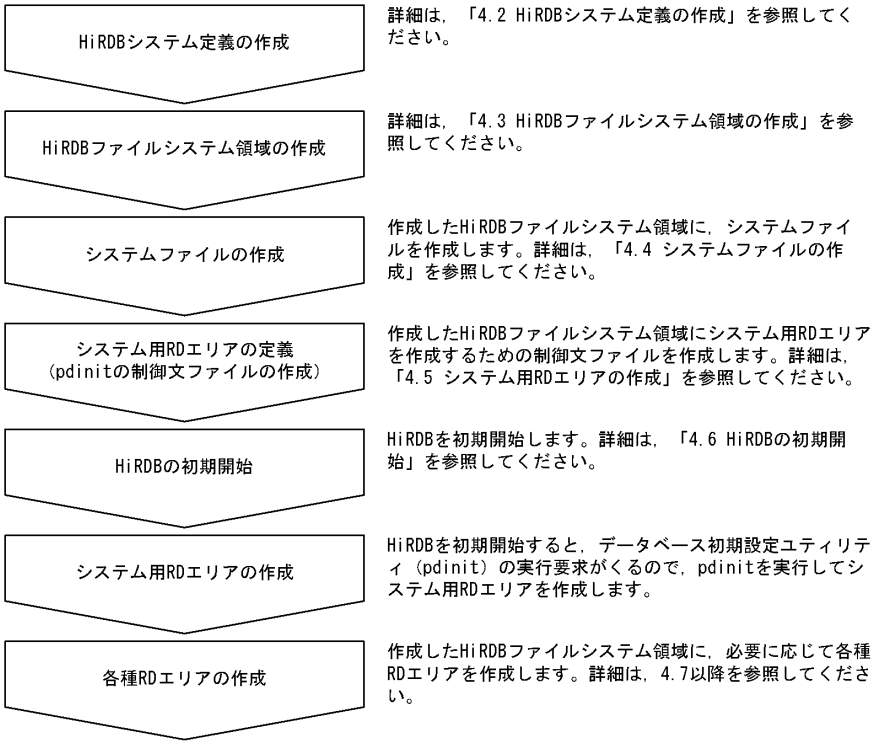
- ユニット及びサーバ構成
- HiRDB ファイルシステム領域の構成
- システムファイルの構成
- 作業表用ファイルの構成
- RD エリアの構成

上記の項目の構成を「8. HiRDB/ シングルサーバの設計」又は「9. HiRDB/ パラレルサーバの設計」を参照して決めてください。その後，4.2 以降の説明に従って HiRDB の環境を設定してください。

## (2) 環境設定手順

コマンドによる HiRDB の環境設定手順を次の図に示します。

**図 4-1 コマンドによる HiRDB の環境設定手順**




以降の説明で設定する内容を次に示します。

**設定する内容**

- データベース初期設定ユティリティ（pdinit）でシステム用 RD エリア（マスタディレクトリ用 RD エリア，データディレクトリ用 RD エリア及びデータディクショナリ用 RD エリア）を作成して，

HiRDB をまず開始できるようにします。

- その後，データベース構成変更ユティリティ（pdmod）で，必要な RD エリア（ユーザ用 RD エリア，データディクショナリ LOB 用 RD エリア，ユーザ LOB 用 RD エリア又はリスト用 RD エリア）を追加します。

なお，ユーザ用 RD エリア，データディクショナリ LOB 用 RD エリア，ユーザ LOB 用 RD エリア及びリスト用 RD エリアは，データベース初期設定ユティリティ（pdinit）でシステム用 RD エリアと一緒に作成することもできます。

# 4.2 HiRDB システム定義の作成

**実行者** HiRDB 管理者

設計したシステム構成及び稼働環境に従って HiRDB システム定義を作成します。ここで説明する項目を次に示します。

- HiRDB システム定義の作成
- HiRDB システム定義の変更方法
- UAP 環境定義の追加又は変更方法

HiRDB システム定義の各オペランドについては，マニュアル「HiRDB Version 8 システム定義」を参照してください。

**留意事項**

- HiRDB システム定義を作成した後に，pdconfchk コマンドで HiRDB システム定義の内容の整合性をチェックしてください。このコマンドは HiRDB を開始するために必要な定義の整合性をチェックします。pdconfchk コマンドでチェックできるオペランドについては，マニュアル「HiRDB Version 8 システム定義」を参照してください。
- HiRDB システム定義ファイルのパーミッションは，ファイルの所有者（HiRDB 管理者）にだけ，読み込み権限及び書き込み権限を持たせるように設定，維持するようにしてください。
- HiRDB システム定義を変更した後に，%PDDIR%¥conf 下のファイルのバックアップを取得してください。HiRDB 運用ディレクトリがあるディスクの障害などに備えて，HiRDB 運用ディレクトリ下のファイル（%PDDIR%¥conf 下のファイル）のバックアップを取得します。HiRDB 運用ディレクトリを回復するには，%PDDIR%¥conf 下のファイルのバックアップが必要になります。

## 4.2.1 HiRDB システム定義の作成（HiRDB/ シングルサーバの場合）

### (1) システム共通定義の作成

システム共通定義には HiRDB の構成及び共通情報を定義します。システム共通定義を作成して次に示すファイルに格納します。

- %PDDIR%¥conf¥pdsys

システム共通定義では，ユニット構成，サーバ構成，及びグローバルバッファの定義をします。

なお，HiRDB のコマンドやユティリティは，この定義ファイルの内容に従って動作するため，HiRDB のコマンド，又はユティリティを実行するユーザ（OS 上のユーザ）に対して，この定義ファイルに対する読み込み権限（r）を与えてください。

### (2) ユニット制御情報定義の作成

ユニット制御情報定義にはユニットの実行環境を定義します。ユニット制御情報定義を作成して次に示すファイルに格納します。

- %PDDIR%¥conf¥pdutsys

ユニット制御情報定義ではユニット用ステータスファイルの定義をします。

なお，HiRDB のコマンドやユティリティは，この定義ファイルの内容に従って動作するため，HiRDB のコマンド，又はユティリティを実行するユーザ（OS 上のユーザ）に対して，この定義ファイルに対する

読み込み権限（r）を与えてください。

### (3) シングルサーバ定義の作成

シングルサーバ定義にはシングルサーバの実行環境を定義します。シングルサーバ定義を作成して次に示すファイルに格納します。

- %PDDIR%¥conf¥ **サーバ名**※

HiRDB のコマンドやユティリティは，この定義ファイルの内容に従って動作するため，HiRDB のコマンド，又はユティリティを実行するユーザ（OS 上のユーザ）に対して，この定義ファイルに対する読み込み権限（r）を与えてください。

シングルサーバ定義で指定する項目の例を次に示します。

- システムログファイル
- シンクポイントダンプファイル
- サーバ用ステータスファイル
- 作業表用ファイル

注※

システム共通定義の pdstart オペランドの -s オプションに指定するサーバ名と同じにしてください。「pdstart -s sds1」と指定した場合は，次に示すファイルに格納してください。

- %PDDIR%¥conf¥sds1

### (4) UAP 環境定義の作成（任意）

UAP の実行環境を定義します。必要に応じて UAP 環境定義を作成して次に示すファイルに格納します。

- %PDDIR%¥conf¥pduapenv¥ **任意の名称**※

なお，HiRDB 管理者は UAP 環境定義を使用するユーザに対して，%PDDIR%¥conf¥pduapenv ディレクトリの読み込み権限（r）と実行権限（x）を与えてください。また，UAP 環境定義ファイルには読み込み権限（r）を与えてください。

また，HiRDB のコマンドやユティリティは，この定義ファイルの内容に従って動作するため，HiRDB のコマンド，又はユティリティを実行するユーザ（OS 上のユーザ）に対して，この定義ファイルに対する読み込み権限（r）を与えてください。

UAP 環境定義で指定する項目の例を次に示します。

- ローカルバッファを使用してアクセスする RD エリア又はインデクスがほかのユーザに使用されている場合の UAP の動作
- UAP が使用するローカルバッファ

注※

ファイル名称は先頭がアルファベットの英数字列（最大 8 文字）にしてください。大文字と小文字は同じ文字とします。例えば，A と a は同じとします。ファイル名 ABC と abc は同じファイル名となります。

### (5) SQL 予約語定義の作成（任意）

SQL 予約語削除機能を使用する場合，UAP ごとに削除する予約語を定義します。必要に応じて SQL 予約語定義を作成して次に示すファイルに格納します。

- %PDDIR%¥conf¥pdrsvwd¥ **任意の名称**※

なお，HiRDB 管理者は SQL 予約語定義を使用するユーザに対して，%PDDIR%**¥**conf**¥**pdrsvwd ディレクトリの読み込み権限（r）と実行権限（x）を与えてください。また，SQL 予約語削除ファイルには読み込み権限（r）を与えてください。

また，HiRDB のコマンドやユティリティは，この定義ファイルの内容に従って動作するため，HiRDB のコマンド，又はユティリティを実行するユーザ（OS 上のユーザ）に対して，この定義ファイルに対する読み込み権限（r）を与えてください。

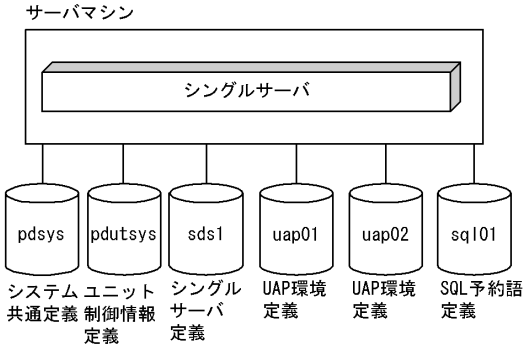
注※

ファイル名称は先頭がアルファベットの英数字列（最大 8 文字）にしてください。大文字と小文字は同じ文字とします。例えば，A と a は同じとします。ファイル名 ABC と abc は同じファイル名となります。

### (6) HiRDB システム定義ファイルの構成例

HiRDB システム定義ファイルの構成例を次の図に示します。

**図** 4-2　HiRDB **システム定義ファイルの構成例（**HiRDB/ **シングルサーバの場合）**




## 4.2.2 HiRDB システム定義の作成（HiRDB/ パラレルサーバの場合）

### (1) システム共通定義の作成

システム共通定義には HiRDB の構成及び共通情報を定義します。システム共通定義を作成して次に示すファイルに格納します。

- %PDDIR%¥conf¥pdsys

各サーバマシンに**同じ内容のシステム共通定義**を作成してください。

システム共通定義では，ユニット構成，サーバ構成，及びグローバルバッファの定義をします。

なお，HiRDB のコマンドやユティリティは，この定義ファイルの内容に従って動作するため，HiRDB のコマンド，又はユティリティを実行するユーザ（OS 上のユーザ）に対して，この定義ファイルに対する読み込み権限（r）を与えてください。

### (2) ユニット制御情報定義の作成

ユニット制御情報定義にはユニットの実行環境を定義します。ユニット制御情報定義を作成して次に示すファイルに格納します。

- %PDDIR%¥conf¥pdutsys

各サーバマシンにユニット制御情報定義を作成してください。

ユニット制御情報定義では，ユニット用ステータスファイルの定義をします。

なお，HiRDB のコマンドやユティリティは，この定義ファイルの内容に従って動作するため，HiRDB のコマンド，又はユティリティを実行するユーザ（OS 上のユーザ）に対して，この定義ファイルに対する読み込み権限（r）を与えてください。

### (3) サーバ共通定義の作成（任意）

サーバ共通定義には，(4) ～ (6) で説明するサーバ定義のオペランドの省略値を定義します。必要に応じて各サーバマシンにサーバ共通定義を作成して次に示すファイルに格納します。

- %PDDIR%¥conf¥pdsvrc

次に示す場合にサーバ共通定義を作成すると便利です。

- 1 **サーバマシンに定義するサーバ数が多い場合**
- **各サーバの定義内容に共通部分が多い場合**

サーバ共通定義で指定した内容は，そのサーバマシンに定義したすべてのサーバに対して有効となります。各サーバの定義内容に共通部分が多い場合は，共通部分をサーバ共通定義で指定し，異なる部分を各サーバ定義で指定することをお勧めします。

また，HiRDB のコマンドやユティリティは，この定義ファイルの内容に従って動作するため，HiRDB のコマンド又はユティリティを実行するユーザ（OS 上のユーザ）に対して，この定義ファイルに対する読み込み権限（r）を与えてください。

### (4) フロントエンドサーバ定義の作成

フロントエンドサーバ定義にはフロントエンドサーバの実行環境を定義します。フロントエンドサーバ定義を作成して次に示すファイルに格納します。

- %PDDIR%¥conf¥ **サーバ名**※

フロントエンドサーバを定義するサーバマシンに，フロントエンドサーバ定義を作成してください。

フロントエンドサーバ定義で指定する項目の例を次に示します。

- フロントエンドサーバ用のシステムログファイル
- フロントエンドサーバ用のシンクポイントダンプファイル
- フロントエンドサーバ用のステータスファイル

注※

システム共通定義の pdstart オペランドの -s オプションに指定するサーバ名と同じにしてください。例えば，「pdstart -s f001」と指定した場合は，次に示すファイルに格納してください。

- %PDDIR%¥conf¥f001

なお，HiRDB のコマンドやユティリティは，この定義ファイルの内容に従って動作するため，HiRDB のコマンド又はユティリティを実行するユーザ（OS 上のユーザ）に対して，この定義ファイルに対する読

み込み権限（r）を与えてください。

## (5) ディクショナリサーバ定義の作成

ディクショナリサーバ定義にはディクショナリサーバの実行環境を定義します。ディクショナリサーバ定義を作成して次に示すファイルに格納します。

- %PDDIR%¥conf¥ **サーバ名**※

ディクショナリサーバを定義するサーバマシンに，ディクショナリサーバ定義を作成してください。

ディクショナリサーバ定義で指定する項目の例を次に示します。

- ディクショナリサーバ用のシステムログファイル
- ディクショナリサーバ用のシンクポイントダンプファイル
- ディクショナリサーバ用のステータスファイル
- 作業表用ファイル

注※

システム共通定義の pdstart オペランドの -s オプションに指定するサーバ名と同じにしてください。例えば，「pdstart -s dic」と指定した場合は，次に示すファイルに格納してください。

- %PDDIR%¥conf¥dic

なお，HiRDB のコマンドやユティリティは，この定義ファイルの内容に従って動作するため，HiRDB のコマンド又はユティリティを実行するユーザ（OS 上のユーザ）に対して，この定義ファイルに対する読み込み権限（r）を与えてください。

## (6) バックエンドサーバ定義の作成

バックエンドサーバ定義にはバックエンドサーバの実行環境を定義します。バックエンドサーバ定義を作成して次に示すファイルに格納します。

- %PDDIR%¥conf¥ **サーバ名**※

バックエンドサーバを定義するサーバマシンに，バックエンドサーバ定義を作成してください。

バックエンドサーバ定義で指定する項目の例を次に示します。

- バックエンドサーバ用のシステムログファイル
- バックエンドサーバ用のシンクポイントダンプファイル
- バックエンドサーバ用のステータスファイル
- 作業表用ファイル

注※

システム共通定義の pdstart オペランドの -s オプションに指定するサーバ名と同じにしてください。例えば，「pdstart -s b001」と指定した場合は，次に示すファイルに格納してください。

- %PDDIR%¥conf¥b001

なお，HiRDB のコマンドやユティリティは，この定義ファイルの内容に従って動作するため，HiRDB のコマンド又はユティリティを実行するユーザ（OS 上のユーザ）に対して，この定義ファイルに対する読み込み権限（r）を与えてください。

## (7) UAP 環境定義の作成（任意）

UAP の実行環境を定義します。必要に応じて UAP 環境定義を作成して次に示すファイルに格納します。

- %PDDIR%¥conf¥pduapenv¥ **任意の名称**※

UAP 環境定義はフロントエンドサーバがあるユニットに作成します。マルチフロントエンドサーバの場合は，UAP 環境定義を適用したいフロントエンドサーバに定義してください。

なお，HiRDB 管理者は UAP 環境定義を使用するユーザに対して，%PDDIR%¥conf¥pduapenv ディレクトリの読み込み権限（r）と実行権限（x）を与えてください。また，UAP 環境定義ファイルには読み込み権限（r）を与えてください。

また，HiRDB のコマンドやユティリティは，この定義ファイルの内容に従って動作するため，HiRDB のコマンド又はユティリティを実行するユーザ（OS 上のユーザ）に対して，この定義ファイルに対する読み込み権限（r）を与えてください。

UAP 環境定義で指定する項目の例を次に示します。

- ローカルバッファを使用してアクセスする RD エリア又はインデクスがほかのユーザに使用されている場合の UAP の動作
- UAP が使用するローカルバッファ

注※

ファイル名称は先頭がアルファベットの英数字列（最大 8 文字）にしてください。大文字と小文字は同じ文字とします。例えば，A と a は同じとします。ファイル名 ABC と abc は同じファイル名となります。

## (8) SQL 予約語定義の作成（任意）

SQL 予約語削除機能を使用する場合，UAP ごとに削除する予約語を定義します。必要に応じて SQL 予約語定義を作成して次に示すファイルに格納します。

- %PDDIR%¥conf¥pdrsvwd¥ **任意の名称**※

SQL 予約語定義はフロントエンドサーバがあるユニットに作成します。マルチフロントエンドサーバの場合は，UAP 環境定義を適用したいフロントエンドサーバに定義してください。

なお，HiRDB 管理者は SQL 予約語定義を使用するユーザに対して，%PDDIR%¥conf¥pdrsvwd ディレクトリの読み込み権限（r）と実行権限（x）を与えてください。また，SQL 予約語削除ファイルには読み込み権限（r）を与えてください。

また，HiRDB のコマンドやユティリティは，この定義ファイルの内容に従って動作するため，HiRDB のコマンド又はユティリティを実行するユーザ（OS 上のユーザ）に対して，この定義ファイルに対する読み込み権限（r）を与えてください。

注※

ファイル名称は先頭がアルファベットの英数字列（最大 8 文字）にしてください。大文字と小文字は同じ文字とします。例えば，A と a は同じとします。ファイル名 ABC と abc は同じファイル名となります。

## (9) 外部サーバ情報定義の作成

HiRDB External Data Access 機能を使用する場合は外部サーバ情報定義が必要です。外部サーバ情報定義には外部サーバへの接続環境を定義します。外部サーバ情報定義を作成して次に示すファイルに格納します。

- %PDDIR%¥conf¥ **外部サーバ名**

外部サーバ接続用のバックエンドサーバを定義するサーバマシンに，外部サーバ情報定義を作成してください。

また，HiRDB のコマンドやユティリティは，この定義ファイルの内容に従って動作するため，HiRDB のコマンド又はユティリティを実行するユーザ（OS 上のユーザ）に対して，この定義ファイルに対する読み込み権限（r）を与えてください。

なお，接続する外部サーバの種類によって外部サーバ情報定義に指定するオペランドが異なります。詳細については，マニュアル「HiRDB Version 8 システム定義」を参照してください。

### （10）Hub 最適化情報定義の作成

HiRDB External Data Access 機能を使用する場合は Hub 最適化情報定義が必要です。Hub 最適化情報定義には HiRDB External Data Access 機能の最適化情報を定義します。Hub 最適化情報定義を作成して次に示すファイルに格納します。

- %PDDIR%¥conf¥ **任意の名称**

フロントエンドサーバを定義するサーバマシンに，Hub 最適化情報定義を作成してください。マルチフロントエンドサーバの場合は，各ユニットに同一内容の Hub 最適化情報定義が必要です。

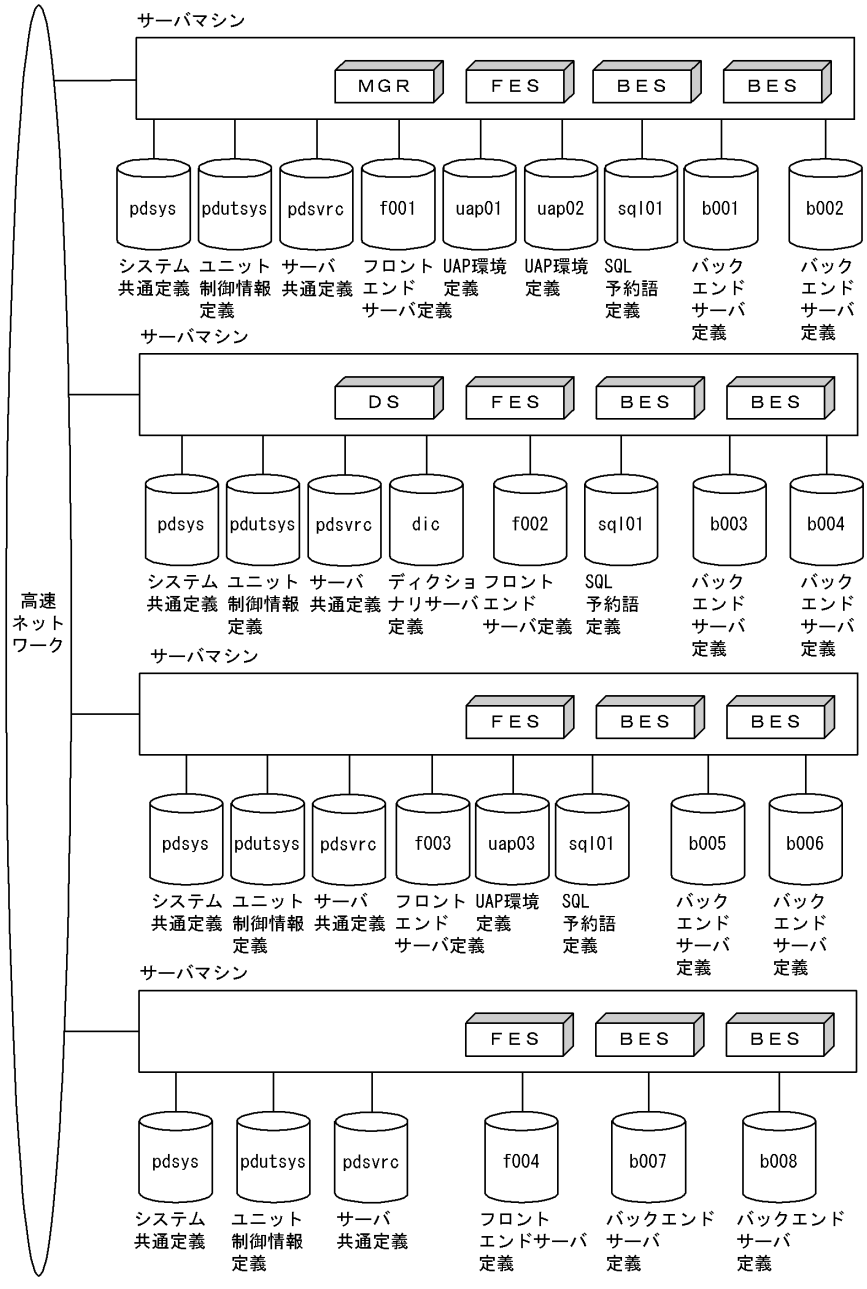
なお，HiRDB のコマンドやユティリティは，この定義ファイルの内容に従って動作するため，HiRDB のコマンド又はユティリティを実行するユーザ（OS 上のユーザ）に対して，この定義ファイルに対する読み込み権限（r）を与えてください。

### （11）HiRDB システム定義ファイルの構成例

HiRDB システム定義ファイルの構成例を次の図に示します。

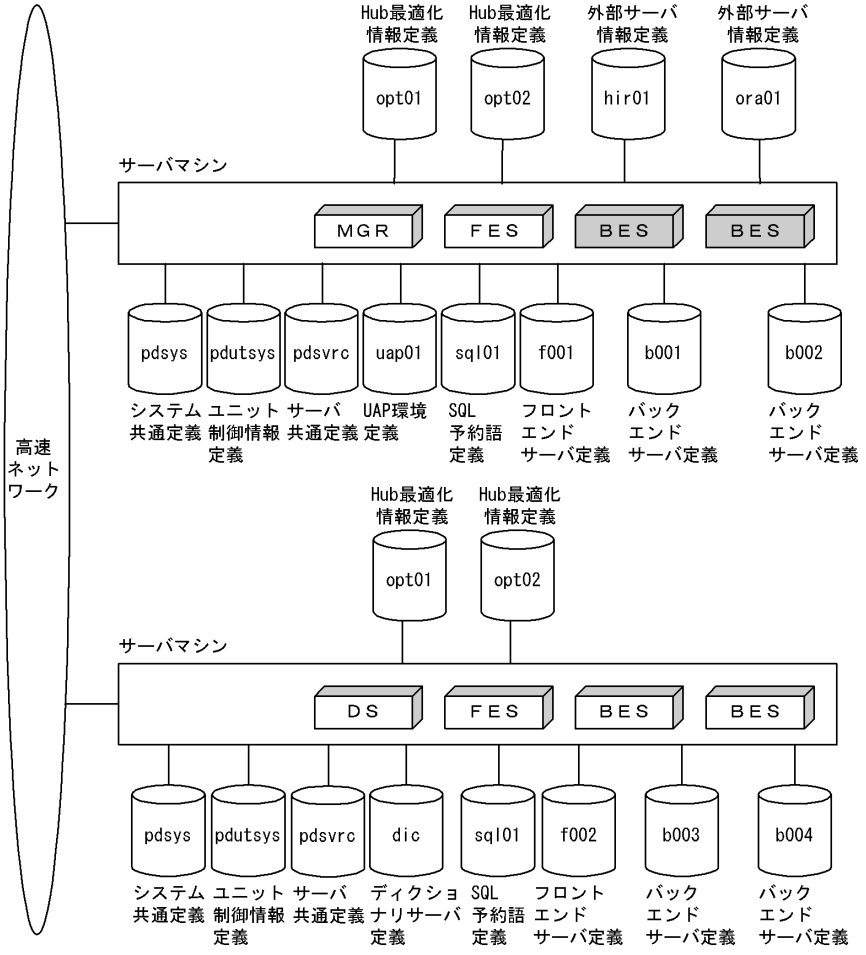
また，HiRDB External Data Access 機能を使用する場合の HiRDB システム定義ファイルの構成例を図 4-4 に示します。

**図 4-3 HiRDB システム定義ファイルの構成例（HiRDB/ パラレルサーバの場合）**




注　各サーバマシンのシステム共通定義は同一内容にしてください。

**図 4-4 HiRDB システム定義ファイルの構成例（HiRDB External Data Access 機能を使用する場合）**




注

- 各サーバマシンのシステム共通定義は同一内容にしてください。
- 網掛けされているバックエンドサーバが外部サーバ接続用のバックエンドサーバです。外部サーバ接続用のバックエンドサーバがあるユニットに外部サーバ情報定義ファイルを作成します。
- マルチフロントエンドサーバ環境のため，各ユニットに Hub 最適化情報定義ファイルを作成します。

## 4.2.3 HiRDB システム定義（UAP 環境定義を除く）の変更方法

HiRDB システム定義を変更する手順を説明します。

**留意事項**

- HiRDB システム定義を変更した後に，%PDDIR%¥conf 下のファイルのバックアップを取得してください。HiRDB 運用ディレクトリがあるディスクの障害などに備えて，HiRDB 運用ディレクトリ下のファイル（%PDDIR%¥conf 下のファイル）のバックアップを取得します。HiRDB 運用ディレ

クトリを回復するには，%PDDIR%¥conf 下のファイルのバックアップが必要になります。また，%PDCONFPATH% が HiRDB 運用ディレクトリ下にある場合は，同様にバックアップを取得してください。

- HiRDB/ パラレルサーバの場合，ユニットごとに %PDDIR%¥conf 及び %PDCONFPATH% 下にサブディレクトリを作成して，HiRDB システム定義の内容をチェックしてください。

### (1) HiRDB システム定義の変更手順

HiRDB システム定義の変更手順を次に示します。なお，%PDDIR%¥conf はユニット制御情報定義ファイルを格納しているディレクトリを意味しています。%PDCONFPATH% はそれ以外の HiRDB システム定義ファイルを格納しているディレクトリを意味しています。

**〈手順〉**

1. %PDDIR%¥conf 及び %PDCONFPATH% 下にサブディレクトリを作成します。この例ではサブディレクトリとして work を作成します。
2. ユニット制御情報定義ファイルを %PDDIR%¥conf¥work 下にコピーします。そのほかの HiRDB システム定義ファイルを %PDCONFPATH%¥work 下にコピーします。
3. %PDDIR%¥conf¥work 及び %PDCONFPATH%¥work 下にコピーした HiRDB システム定義を変更します。
4. pdconfchk -d work コマンドで，%PDDIR%¥conf¥work 及び %PDCONFPATH%¥work 下の HiRDB システム定義の内容をチェックします。エラーがある場合は HiRDB システム定義を修正して，再度 pdconfchk コマンドを実行してください。
5. pdstop コマンドで HiRDB を正常終了します。
6. pdlogunld コマンドで，アンロード待ち状態のシステムログファイルをアンロードします。
7. 3 で変更した HiRDB システム定義ファイルを %PDDIR%¥conf 又は %PDCONFPATH% 下にコピーして，HiRDB システム定義ファイルを置き換えます。
8. 次に示すオペランドの指定値を変更した場合は，pdloginit コマンドでシステムログファイルを初期化します。
   - pd_log_dual
   - pdstart
9. pdstart コマンドで HiRDB を正常開始します。

### (2) システム構成変更コマンドを使用した HiRDB システム定義の変更手順

システム構成変更コマンド（pdchgconf コマンド）を使用すると，HiRDB の稼働中に HiRDB システム定義を変更できるため，HiRDB を終了する必要はありません。ただし，このコマンドを使用する場合は HiRDB Advanced High Availability が必要になります。システム構成変更コマンドを使用した HiRDB システム定義の変更手順を次に示します。

**〈手順〉**

1. %PDDIR%¥conf¥chgconf ディレクトリを作成します。
2. 使用中の HiRDB システム定義ファイルを 1 で作成したディレクトリ下にコピーします。
3. %PDDIR%¥conf¥chgconf 下の HiRDB システム定義を変更します。
4. pdconfchk コマンドで，%PDDIR%¥conf¥chgconf 下の HiRDB システム定義のチェックを行います。エラーがある場合は HiRDB システム定義を修正して，再度 pdconfchk コマンドを実行してください。
5. pdchgconf コマンドで，HiRDB システム定義を変更後の HiRDB システム定義に置き換えます。pdchgconf コマンドを実行すると，使用中（変更前）の HiRDB システム定義ファイルが %PDDIR%¥conf¥backconf 下に退避されます。そして，%PDDIR%¥conf¥chgconf 下の変更後の

HiRDB システム定義ファイルが %PDDIR%¥conf 下にコピーされます。

**注意事項**

- pdchgconf コマンドの入力後，15 分以上トランザクション又はユティリティが動き続けた場合，pdchgconf コマンドが異常終了します。
- システム構成変更コマンドを使用した HiRDB システム定義の変更には制限事項があります。制限事項については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

### (3) 注意事項

- システム共通定義を修正する場合は，すべてのサーバマシンのシステム共通定義を同じように修正してください（HiRDB/ パラレルサーバの場合）。
- 稼働中の HiRDB が使用している HiRDB システム定義は，変更又は削除しないでください。変更又は削除した場合，その HiRDB の動作は保証できません。
- HiRDB が計画停止，強制終了，又は異常終了した場合，HiRDB システム定義のオペランドで変更できるものと変更できないものがあります。詳細については，マニュアル「HiRDB Version 8 システム定義」を参照してください。

## 4.2.4 UAP 環境定義の追加又は変更方法

UAP 環境定義の追加又は変更手順を次に示します。

**〈手順〉**

1. UAP 環境定義を使用する UAP が実行中でないか確認します。UAP の実行中に UAP 環境定義を追加又は変更すると，実行中の UAP は変更前の UAP 環境定義が適用されます。ただし，タイミングによっては変更後の UAP 環境定義が適用されることがあります。
2. UAP 環境定義を追加又は変更します。
3. 追加又は変更した UAP 環境定義を使用して UAP を実行します。

# 4.3 HiRDB ファイルシステム領域の作成

**実行者** HiRDB 管理者

HiRDB ファイルを作成する領域（HiRDB **ファイルシステム領域**）を pdfmkfs コマンドで作成します。

## 4.3.1 HiRDB ファイルシステム領域の種類

HiRDB ファイルシステム領域は，次の表に示す**用途ごとに**作成してください。用途は pdfmkfs コマンドの -k オプションで指定します。

**表 4-1 HiRDB ファイルシステム領域の種類**

| 項番 | HiRDB ファイルシステム領域の種類 | -k オプションの指定 |
|---|---|---|
| 1 | RD エリア用 | DB |
| 2 | 共用 RD エリア用 | SDB |
| 3 | システムファイル用 | SYS |
| 4 | 作業表用ファイル用 | WORK |
| 5 | ユティリティ用 | UTL |
| 6 | ユティリティ用（Windows のキャッシュを使用しない） | NUTL |
| 7 | リスト用 RD エリア用 | WORK |

HiRDB を稼働するためには，1，3，及び 4 の HiRDB ファイルシステム領域が必要です。

HiRDB ファイルシステム領域の設計方法については，HiRDB/ シングルサーバの場合は，「8.2 HiRDB ファイルシステム領域の設計」，HiRDB/ パラレルサーバの場合は，「9.2 HiRDB ファイルシステム領域の設計」を参照してください。

**注意事項**

作成する HiRDB ファイルシステムの領域長は，パーティションの領域長と等しいか又は小さくしてください。パーティションの領域長より大きくすると，そのパーティションに物理的に続くパーティションを破壊する場合があります。

**HiRDB ファイルシステム領域の最大長**

HiRDB ファイルシステム領域の最大長は次のとおりです。

| 条件 | HiRDB ファイルシステム領域の最大長 |
|---|---|
| pd_large_file_use ＝ N 指定時 | 2,047 メガバイト |
| pd_large_file_use ＝ Y 指定時（省略値） | 1,048,575 メガバイト |

## 4.3.2 raw I/O 機能を使用した HiRDB ファイルシステム領域の作成

パーティション又は論理ドライブをファイルと同様にアクセスする Windows のダイレクトディスクアクセス（raw I/O）を使用する HiRDB ファイルシステム領域を作成できます。この機能を raw I/O **機能**といいます。raw I/O 機能を使用すると，HiRDB は Windows のファイルキャッシュの動作による影響を受けなくなります。そのため，グローバルバッファ制御などによって安定した性能を維持できるようになります。

## (1) raw I/O 機能の適用範囲

raw I/O 機能を適用できる HiRDB ファイルシステム領域のサポート範囲を次に示します。

| 使用目的※1 | サポートの有無 |
|---|---|
| DB | ○ |
| SDB | ○ |
| SYS | ○ |
| WORK | ○ |
| UTL | × |
| SVR | ○ |
| NUTL※2 | ○ |

（凡例）

○：raw I/O 機能を適用できます。

×：raw I/O 機能を適用できません。

注※ 1

pdfmkfs コマンドの -k オプションで指定する HiRDB ファイルシステム領域の使用目的です。

注※ 2

raw I/O 機能を適用できるユティリティ用ファイルを次に示します。

- バックアップファイル
- アンロードログファイル
- アンロードデータファイル
- 差分バックアップ管理ファイル
- インデクス情報ファイル

## (2) raw I/O 機能を使用するための準備

raw I/O 機能を使用する場合，pdfmkfs コマンドを実行する前に，ここで説明する準備が必要です。

### (a) ディスクについて

raw I/O 機能では，1 ドライブを 1HiRDB ファイルシステム領域に割り当てます。このとき，1 ドライブは 1 ディスクで構成されている必要があります。1 ディスクは，OS から一つの記憶域として見えればよいため，ハードウェア RAID は使用できますが，1 ドライブを複数ディスクで構成するソフトウェア RAID は使用できません。

なお，raw I/O 機能を使用できるドライブは，セクタ長 512 バイトの固定ディスクだけです。

ディスクの種類とパーティションスタイルの組み合わせによる HiRDB のサポート状況を次に示します。

| ディスクの種類 | パーティションスタイル | サポート状況 | |
|---|---|---|---|
| | | 32 ビットモードの HiRDB | 64 ビットモードの HiRDB |
| ベーシックディスク | MBR（マスタブートレコード） | ○ | ○ |
| | GPT（GUID パーティションテーブル） | × | ○ |

| ディスクの種類 | パーティションスタイル | サポート状況 | |
|---|---|---|---|
| | | 32 ビットモードの HiRDB | 64 ビットモードの HiRDB |
| ダイナミックディスク | MBR（マスタブートレコード） | × | × |
| | GPT（GUID パーティションテーブル） | × | × |

（凡例）
○：サポートしています。
×：サポートしていません。

**(b) パーティション及び論理ドライブの作成方法**

raw I/O 機能を使用するには，未フォーマット状態のパーティション，又は論理ドライブを用意します。このとき，1 ドライブは，1 プライマリパーティション，又は 1 論理ドライブで構成されている必要があります。

パーティションは Windows の［コンピュータの管理］－［ディスクの管理］で作成します。パーティションのサイズには，HiRDB ファイルシステム領域のサイズより大きい値を指定してください。ただし，パーティションのサイズを大きくし過ぎると，むだな領域ができますので，注意してください。

また，作成したパーティションには，ドライブ文字を割り当ててください。

パーティション，又は論理ドライブの作成方法，及びドライブ文字の割り当て方法の詳細については，Windows の［ディスクの管理］のヘルプを参照してください。

### 4.3.3 例題 1（RD エリア用の HiRDB ファイルシステム領域の作成）

RD エリア用の HiRDB ファイルシステム領域を作成する例を次に示します。

（例）
RD エリア用の HiRDB ファイルシステム領域を作成します。

```
pdfmkfs -n 50 -l 10 -k DB -i C:¥dbarea01
```

〔説明〕

-n：HiRDB ファイルシステム領域の領域長をメガバイト単位で指定します。

-l：HiRDB ファイルシステム領域内に作成する HiRDB ファイル数の上限値を指定します。

-k：HiRDB ファイルシステム領域の用途を指定します。
RD エリア用の HiRDB ファイルシステム領域なので，DB を指定します。

-i：HiRDB ファイルシステム領域の全領域を初期化する場合に指定します。
-i オプションを指定すると，領域全体を確保します。-i オプションを省略すると，HiRDB ファイルシステム領域の管理情報だけを作成します。

C:¥dbarea01：作成する HiRDB ファイルシステム領域の名称を指定します。

コマンドの実行後，実行結果が正しいかどうかを確認することをお勧めします。コマンドの実行結果の確認方法については，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

### 4.3.4 例題 2（システムファイル用の HiRDB ファイルシステム領域の作成）

システムファイル用の HiRDB ファイルシステム領域を作成する例を次に示します。

（例）

システムファイル用の HiRDB ファイルシステム領域を作成します。

```
pdfmkfs -n 50 -l 20 -k SYS -i C:¥sysarea01
```

〔説明〕

-n：HiRDB ファイルシステム領域の領域長をメガバイト単位で指定します。

-l：HiRDB ファイルシステム領域内に作成する HiRDB ファイル数の上限値を指定します。

-k：HiRDB ファイルシステム領域の用途を指定します。
システムファイル用の HiRDB ファイルシステム領域なので，SYS を指定します。

-i：HiRDB ファイルシステム領域の全領域を初期化する場合に指定します。
-i オプションを指定すると，領域全体を確保します。-i オプションを省略すると，HiRDB ファイルシステム領域の管理情報だけを作成します。

C:¥sysarea01：作成する HiRDB ファイルシステム領域の名称を指定します。

コマンドの実行後，実行結果が正しいかどうかを確認することをお勧めします。コマンドの実行結果の確認方法については，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

### 4.3.5 例題 3（作業表用ファイル用の HiRDB ファイルシステム領域の作成）

作業表用ファイル用の HiRDB ファイルシステム領域を作成する例を次に示します。

（例）

作業表用ファイル用の HiRDB ファイルシステム領域を作成します。

```
pdfmkfs -n 50 -l 20 -k WORK -e 3300 -i -a C:¥workarea01
```

〔説明〕

-n：HiRDB ファイルシステム領域の領域長をメガバイト単位で指定します。
領域長の見積もり方法については，「18. 作業表用ファイルの容量の見積もり」を参照してください。

-l：HiRDB ファイルシステム領域内に作成する HiRDB ファイル数の上限値を指定します。

-k：HiRDB ファイルシステム領域の用途を指定します。
作業表用ファイル用の HiRDB ファイルシステム領域なので，WORK を指定します。

-e：HiRDB ファイルシステム領域内の HiRDB ファイルの増分回数を指定します。

-i：HiRDB ファイルシステム領域の全領域を初期化する場合に指定します。
-i オプションを指定すると，領域全体を確保します。-i オプションを省略すると，HiRDB ファイルシステム領域の管理情報だけを作成します。

-a：自動的に HiRDB ファイルシステム領域を拡張するときに指定します。
RD エリアの自動増分や作業表を使用する SQL の実行などで，-n オプションで指定したサイズを超えても，自動的に必要な分だけ HiRDB ファイルシステム領域を拡張するときに指定します。

C:¥workarea01：
作成する HiRDB ファイルシステム領域の名称を指定します。**HiRDB システム定義の** pdwork **オペランドで指定した名称を指定します。**

コマンドの実行後，実行結果が正しいかどうかを確認することをお勧めします。コマンドの実行結果の確認方法については，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

### 4.3.6 例題 4（ユティリティ用の HiRDB ファイルシステム領域の作成）

ユティリティ用の HiRDB ファイルシステム領域を作成する例を次に示します。ユティリティ用の HiRDB ファイルシステム領域には，次に示すファイルを作成します。

- バックアップファイル
- アンロードデータファイル
- アンロードログファイル
- 差分バックアップ管理ファイル
- インデクス情報ファイル

（例）
ユティリティ用の HiRDB ファイルシステム領域を作成します。

```
pdfmkfs -n 50 -l 10 -k UTL -i C:¥utlarea01
```

〔説明〕

-n：HiRDB ファイルシステム領域の領域長をメガバイト単位で指定します。

-l：HiRDB ファイルシステム領域内に作成する HiRDB ファイル数の上限値を指定します。

-k：HiRDB ファイルシステム領域の用途を指定します。
ユティリティ用の HiRDB ファイルシステム領域なので，UTL を指定します。

-i：HiRDB ファイルシステム領域の全領域を初期化する場合に指定します。
-i オプションを指定すると，領域全体を確保します。-i オプションを省略すると，HiRDB ファイルシステム領域の管理情報だけを作成します。

C:¥utlarea01：作成する HiRDB ファイルシステム領域の名称を指定します。

コマンドの実行後，実行結果が正しいかどうかを確認することをお勧めします。コマンドの実行結果の確認方法については，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

### 4.3.7 例題 5（リスト用 RD エリア用の HiRDB ファイルシステム領域の作成）

リスト用 RD エリア用の HiRDB ファイルシステム領域を作成する例を次に示します。

（例）
リスト用 RD エリア用の HiRDB ファイルシステム領域を作成します。

```
pdfmkfs -n 50 -l 10 -k WORK -i C:¥listarea01
```

〔説明〕

-n：HiRDB ファイルシステム領域の領域長をメガバイト単位で指定します。

-l：HiRDB ファイルシステム領域内に作成する HiRDB ファイル数の上限値を指定します。

-k：HiRDB ファイルシステム領域の用途を指定します。
リスト用 RD エリア用の HiRDB ファイルシステム領域なので，WORK を指定します。

-i：HiRDB ファイルシステム領域の全領域を初期化する場合に指定します。
-i オプションを指定すると，領域全体を確保します。-i オプションを省略すると，HiRDB ファイルシステム領域の管理情報だけを作成します。

C:¥listarea01：作成する HiRDB ファイルシステム領域の名称を指定します。

コマンドの実行後，実行結果が正しいかどうかを確認することをお勧めします。コマンドの実行結果の確認方法については，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

### 4.3.8 例題 6（raw I/O 機能を使用する HiRDB ファイルシステム領域の作成）

raw I/O 機能を使用する HiRDB ファイルシステム領域を作成するには，次に示す形式で HiRDB ファイルシステム領域名を指定します。

**¥¥.¥ ドライブ名**：
ドライブ名には，事前に準備したドライブ文字を指定します。

（例）
RD エリア用の HiRDB ファイルシステム領域を作成します。

```
pdfmkfs -n 100 -l 50 -e 1 -k DB ¥¥.¥J:
```

〔説明〕

-n：HiRDB ファイルシステム領域の領域長をメガバイト単位で指定します。

-l：HiRDB ファイルシステム領域内に作成する HiRDB ファイル数の上限値を指定します。

-e：HiRDB ファイルシステム領域内の HiRDB ファイルの増分回数を指定します。

-k：HiRDB ファイルシステム領域の用途を指定します。
RD エリア用の HiRDB ファイルシステム領域なので，DB を指定します。

¥¥.¥J：論理ディスク J を HiRDB ファイルシステム領域として作成します。

# 4.4 システムファイルの作成

**実行者　HiRDB 管理者**

「4.3　HiRDB ファイルシステム領域の作成」で作成したシステムファイル用の HiRDB ファイルシステム領域に，次に示すファイル（システムファイル）を作成します。

- システムログファイル
- シンクポイントダンプファイル
- ステータスファイル

システムファイルの設計方法については，HiRDB/ シングルサーバの場合は「8.3　システムファイルの設計」，HiRDB/ パラレルサーバの場合は「9.3　システムファイルの設計」を参照してください。

## 4.4.1 システムログファイルの作成

pdloginit コマンドで，HiRDB ファイルシステム領域にシステムログファイルを作成します。

（例）

HiRDB ファイルシステム領域（C:¥sysarea01）にシステムログファイル（log01）を作成します。

```
pdloginit -d sys -s b001 -f C:¥sysarea01¥log01 -n 1024
```

〔説明〕

-d sys：システムログファイルを作成する場合に指定します。

-s：システムログファイルを作成するサーバの名称を指定します。
HiRDB/ シングルサーバの場合は指定は不要です。

-f：システムログファイルの名称を指定します。
HiRDB システム定義のサーバ定義の pdlogadpf -d sys オペランドで指定した名称を指定します。

-n：システムログファイルのレコード数を指定します。
1 システムログファイルの容量はレコード長×レコード数（バイト）になります。システムログファイルのレコード長は通常 4096 バイトですが，pd_log_rec_leng オペランドで変更できます。

コマンドの実行後，実行結果が正しいかどうか確認することをお勧めします。コマンドの実行結果の確認方法については，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

**HiRDB システム定義との関係**

HiRDB システム定義のサーバ定義の次に示すオペランドと関係があります。

- pdlogadfg -d sys
- pdlogadpf -d sys

作成したシステムログファイルは，これらのオペランドで定義しておく必要があります。

## 4.4.2 シンクポイントダンプファイルの作成

pdloginit コマンドで，HiRDB ファイルシステム領域にシンクポイントダンプファイルを作成します。

（例）

HiRDB ファイルシステム領域（C:¥sysarea01）にシンクポイントダンプファイル（sync01）を作成します。

```
pdloginit -d spd -s b001 -f C:¥sysarea01¥sync01 -n 64
```

〔説明〕

-d spd：シンクポイントダンプファイルを作成する場合に指定します。

-s：シンクポイントダンプファイルを作成するサーバの名称を指定します。
HiRDB/ シングルサーバの場合は指定は不要です。

-f：シンクポイントダンプファイルの名称を指定します。
HiRDB システム定義のサーバ定義の pdlogadpf -d spd オペランドで指定した名称を指定します。

-n：シンクポイントダンプファイルのレコード数を指定します。
1 シンクポイントダンプファイルの容量は 4096 ×レコード数（バイト）になります。

コマンドの実行後，実行結果が正しいかどうか確認することをお勧めします。コマンドの実行結果の確認方法については，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

**HiRDB システム定義との関係**

HiRDB システム定義のサーバ定義の次に示すオペランドと関係があります。

- pdlogadfg -d spd
- pdlogadpf -d spd

作成したシンクポイントダンプファイルは，これらのオペランドで定義しておく必要があります。

## 4.4.3 ステータスファイルの作成

pdstsinit コマンドで，HiRDB ファイルシステム領域にステータスファイルを作成します。ステータスファイルは，ユニット用ステータスファイルとサーバ用ステータスファイルの両方を作成します。

（例）

HiRDB ファイルシステム領域（C:¥sysarea01）にサーバ用ステータスファイル（sts01）を作成します。

```
pdstsinit -s b001 -f C:¥sysarea01¥sts01 -l 4096 -c 256
```

〔説明〕

-s：サーバ用ステータスファイルを作成するサーバの名称を指定します。

-f：サーバ用ステータスファイルの名称を指定します。
HiRDB システム定義のサーバ定義の pd_sts_file_name オペランドで指定した名称を指定します。

-l：ステータスファイルのレコード長を指定します。

-c：ステータスファイルのレコード数を指定します。
1 ステータスファイルの容量はレコード長×レコード数（バイト）になります。

コマンドの実行後，実行結果が正しいかどうか確認することをお勧めします。コマンドの実行結果の確認方法については，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

**HiRDB システム定義との関係**

HiRDB システム定義の次に示すオペランドと関係があります。

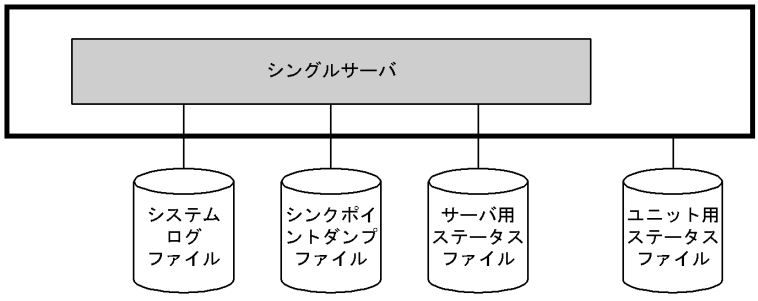
- pd_syssts_file_name（ユニット用ステータスファイル）

- pd_sts_file_name（サーバ用ステータスファイル）

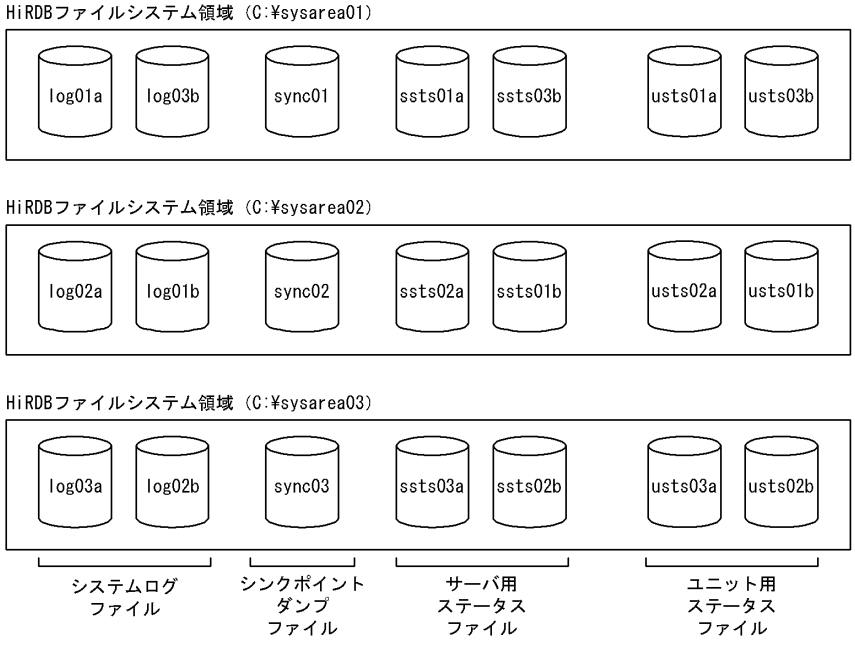
作成したステータスファイルは，これらのオペランドで定義しておく必要があります。

## 4.4.4 システムファイルの作成例（HiRDB/ シングルサーバの場合）

次に示すシステム構成のシステムファイルの作成例を説明します。




HiRDB ファイルシステム領域の構成




### （1） システムファイルの定義（HiRDB システム定義の指定）

HiRDB システム定義にシステムファイルを定義します。

#### （a） ユニット制御情報定義（ユニット用ステータスファイルの定義）

ユニット制御情報定義にユニット用ステータスファイルを定義します。

**定義例**

```
set pd_syssts_file_name_1="usts1","C:¥sysarea01¥usts01a"¥
                                 ,"C:¥sysarea02¥usts01b"
set pd_syssts_file_name_2="usts2","C:¥sysarea02¥usts02a"¥
                                 ,"C:¥sysarea03¥usts02b"
set pd_syssts_file_name_3="usts3","C:¥sysarea03¥usts03a"¥
                                 ,"C:¥sysarea01¥usts03b"
```

#### (b) シングルサーバ定義

シングルサーバ定義にシステムログファイル，シンクポイントダンプファイル，及びサーバ用ステータスファイルを定義します。

**システムログファイルの定義例**

```
set pd_log_rec_leng=4096
pdlogadfg -d sys -g log1 ONL
pdlogadfg -d sys -g log2 ONL
pdlogadfg -d sys -g log3 ONL
pdlogadpf -d sys -g log1 -a "C:¥sysarea01¥log01a"¥
                         -b "C:¥sysarea02¥log01b"
pdlogadpf -d sys -g log2 -a "C:¥sysarea02¥log02a"¥
                         -b "C:¥sysarea03¥log02b"
pdlogadpf -d sys -g log3 -a "C:¥sysarea03¥log03a"¥
                         -b "C:¥sysarea01¥log03b"
```

**シンクポイントダンプファイルの定義例**

```
pdlogadfg -d spd -g sync1 ONL
pdlogadfg -d spd -g sync2 ONL
pdlogadfg -d spd -g sync3 ONL
pdlogadpf -d spd -g sync1 -a "C:¥sysarea01¥sync01"
pdlogadpf -d spd -g sync2 -a "C:¥sysarea02¥sync02"
pdlogadpf -d spd -g sync3 -a "C:¥sysarea03¥sync03"
```

**サーバ用ステータスファイルの定義例**

```
set pd_sts_file_name_1="ssts1","C:¥sysarea01¥ssts01a"¥
                               ,"C:¥sysarea02¥ssts01b"
set pd_sts_file_name_2="ssts2","C:¥sysarea02¥ssts02a"¥
                               ,"C:¥sysarea03¥ssts02b"
set pd_sts_file_name_3="ssts3","C:¥sysarea03¥ssts03a"¥
                               ,"C:¥sysarea01¥ssts03b"
```

### (2) HiRDB ファイルシステム領域の作成

pdfmkfs コマンドで HiRDB ファイルシステム領域を作成します。

**コマンドの入力例**

```
pdfmkfs -n 50 -l 20 -i -k SYS C:¥sysarea01
pdfmkfs -n 50 -l 20 -i -k SYS C:¥sysarea02
pdfmkfs -n 50 -l 20 -i -k SYS C:¥sysarea03
```

### (3) システムファイルの作成

#### (a) システムログファイルの作成

pdloginit コマンドでシステムログファイルを作成します。

**コマンドの入力例**

```
pdloginit -d sys -f C:¥sysarea01¥log01a -n 1024
pdloginit -d sys -f C:¥sysarea01¥log03b -n 1024
pdloginit -d sys -f C:¥sysarea02¥log02a -n 1024
pdloginit -d sys -f C:¥sysarea02¥log01b -n 1024
pdloginit -d sys -f C:¥sysarea03¥log03a -n 1024
pdloginit -d sys -f C:¥sysarea03¥log02b -n 1024
```

#### (b) シンクポイントダンプファイルの作成

pdloginit コマンドでシンクポイントダンプファイルを作成します。

**コマンドの入力例**

```
pdloginit -d spd -f C:¥sysarea01¥sync01 -n 64
pdloginit -d spd -f C:¥sysarea02¥sync02 -n 64
pdloginit -d spd -f C:¥sysarea03¥sync03 -n 64
```

#### (c) サーバ用ステータスファイルの作成

pdstsinit コマンドでサーバ用ステータスファイルを作成します。

**コマンドの入力例**

```
pdstsinit -s sds1 -f C:¥sysarea01¥ssts01a -l 4096 -c 256
pdstsinit -s sds1 -f C:¥sysarea01¥ssts03b -l 4096 -c 256
pdstsinit -s sds1 -f C:¥sysarea02¥ssts02a -l 4096 -c 256
pdstsinit -s sds1 -f C:¥sysarea02¥ssts01b -l 4096 -c 256
pdstsinit -s sds1 -f C:¥sysarea03¥ssts03a -l 4096 -c 256
pdstsinit -s sds1 -f C:¥sysarea03¥ssts02b -l 4096 -c 256
```

#### (d) ユニット用ステータスファイルの作成

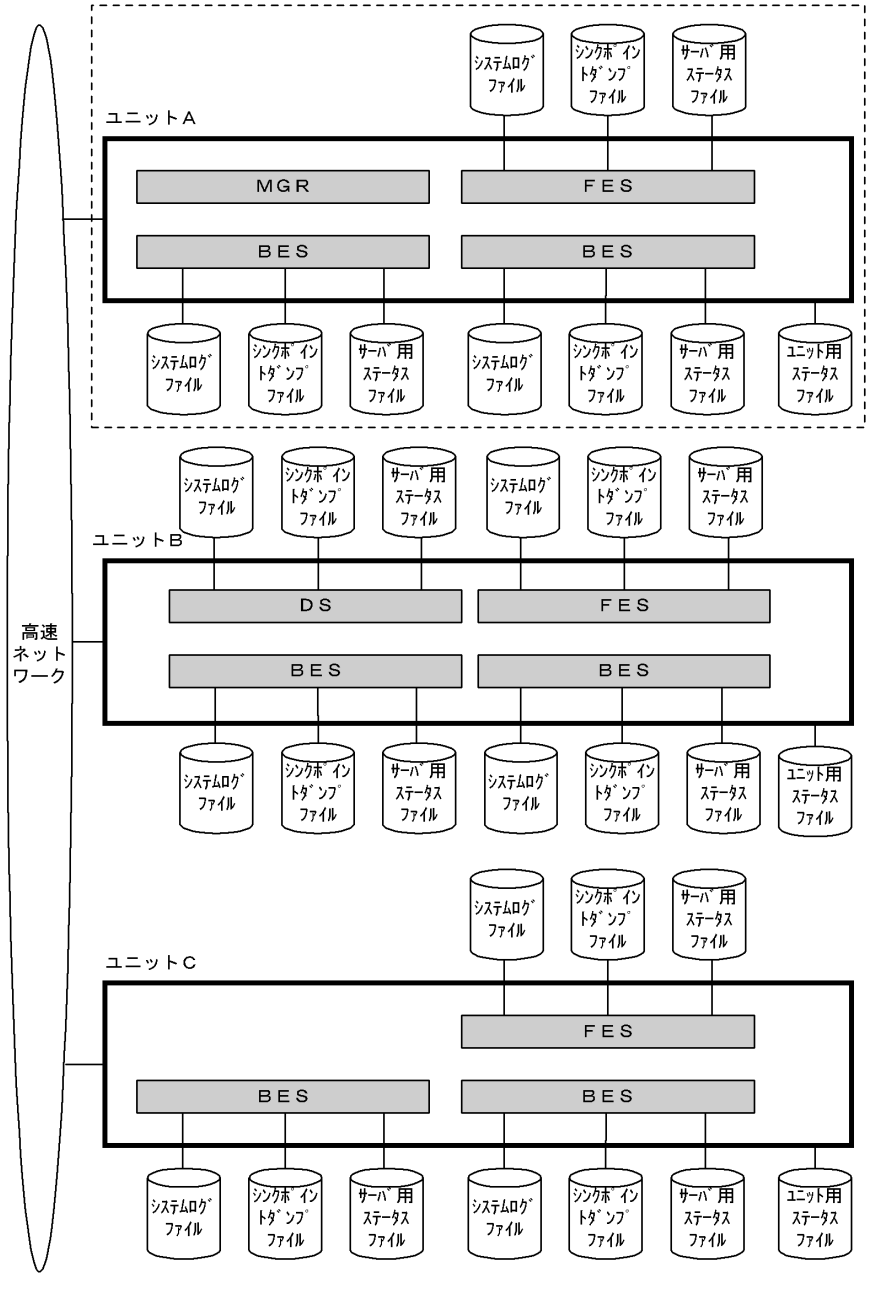
pdstsinit コマンドでユニット用ステータスファイルを作成します。

**コマンドの入力例**

```
pdstsinit -u unt1 -f C:¥sysarea01¥usts01a -l 4096 -c 256
pdstsinit -u unt1 -f C:¥sysarea01¥usts03b -l 4096 -c 256
pdstsinit -u unt1 -f C:¥sysarea02¥usts02a -l 4096 -c 256
pdstsinit -u unt1 -f C:¥sysarea02¥usts01b -l 4096 -c 256
pdstsinit -u unt1 -f C:¥sysarea03¥usts03a -l 4096 -c 256
pdstsinit -u unt1 -f C:¥sysarea03¥usts02b -l 4096 -c 256
```

## 4.4.5 システムファイルの作成例（HiRDB/ パラレルサーバの場合）

次に示すシステム構成のシステムファイルの作成例を説明します。




（凡例）

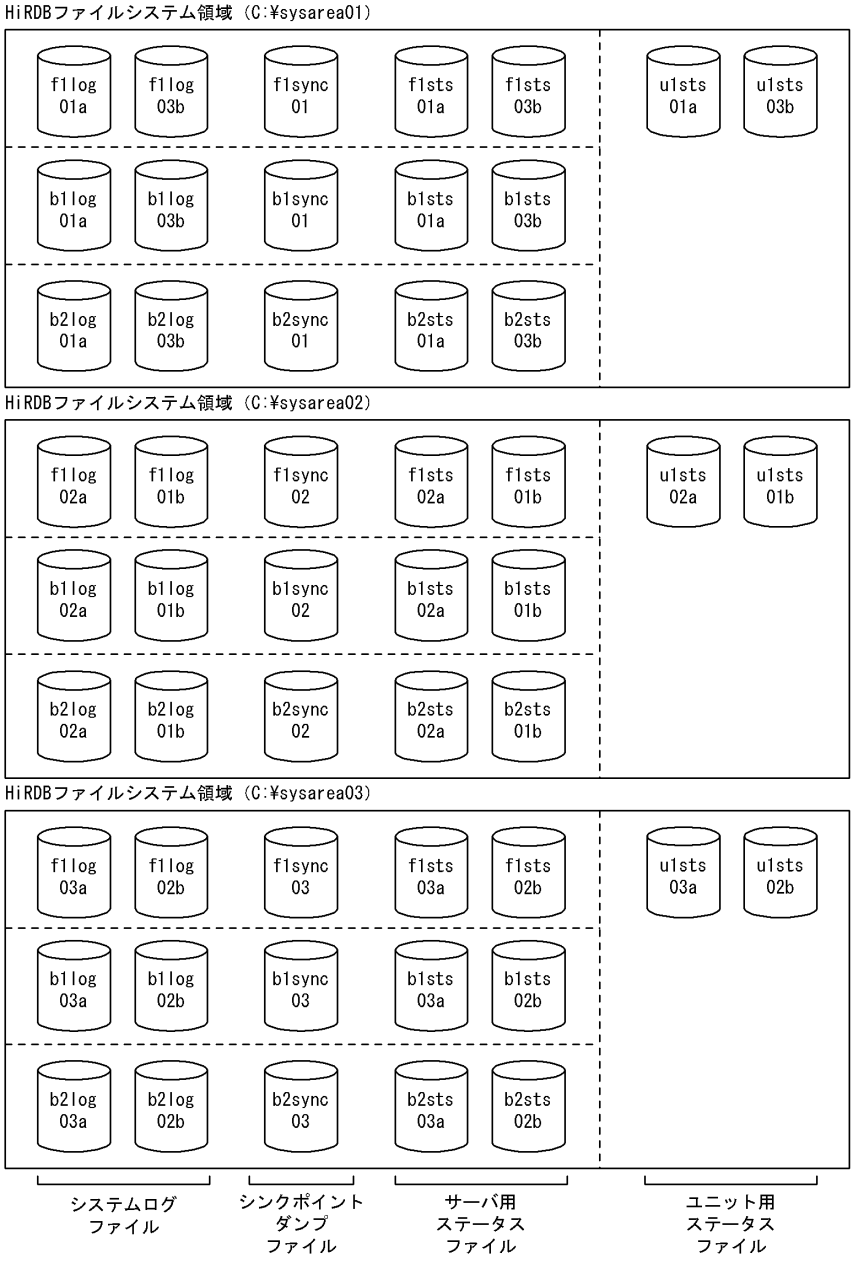
MGR：システムマネジャ

FES：フロントエンドサーバ

DS：ディクショナリサーバ

BES：バックエンドサーバ

HiRDB **ファイルシステム領域の構成**




〔説明〕

ユニット A の HiRDB ファイルシステム領域の構成例です。以降の例題では，ユニット A のシステムファイルの作成例についてだけ説明します。

## (1) システムファイルの定義（HiRDB システム定義の指定）

HiRDB システム定義にシステムファイルを定義します。

### (a) ユニット制御情報定義（ユニット用ステータスファイルの定義）

ユニット制御情報定義にユニット用ステータスファイルを定義します。

**定義例**

```
set pd_syssts_file_name_1="u1sts1","C:¥sysarea01¥u1sts01a"¥
                                  ,"C:¥sysarea02¥u1sts01b"
set pd_syssts_file_name_2="u1sts2","C:¥sysarea02¥u1sts02a"¥
                                  ,"C:¥sysarea03¥u1sts02b"
set pd_syssts_file_name_3="u1sts3","C:¥sysarea03¥u1sts03a"¥
                                  ,"C:¥sysarea01¥u1sts03b"
```

### (b) FES1 のフロントエンドサーバ定義

FES1 のフロントエンドサーバ定義にシステムログファイル，シンクポイントダンプファイル，及びサーバ用ステータスファイルを定義します。

**システムログファイルの定義例**

```
set pd_log_rec_leng=4096
pdlogadfg -d sys -g f1log1 ONL
pdlogadfg -d sys -g f1log2 ONL
pdlogadfg -d sys -g f1log3 ONL
pdlogadpf -d sys -g f1log1 -a "C:¥sysarea01¥f1log01a"¥
                           -b "C:¥sysarea02¥f1log01b"
pdlogadpf -d sys -g f1log2 -a "C:¥sysarea02¥f1log02a"¥
                           -b "C:¥sysarea03¥f1log02b"
pdlogadpf -d sys -g f1log3 -a "C:¥sysarea03¥f1log03a"¥
                           -b "C:¥sysarea01¥f1log03b"
```

**シンクポイントダンプファイルの定義例**

```
pdlogadfg -d spd -g f1sync1 ONL
pdlogadfg -d spd -g f1sync2 ONL
pdlogadfg -d spd -g f1sync3 ONL
pdlogadpf -d spd -g f1sync1 -a "C:¥sysarea01¥f1sync01"
pdlogadpf -d spd -g f1sync2 -a "C:¥sysarea02¥f1sync02"
pdlogadpf -d spd -g f1sync3 -a "C:¥sysarea03¥f1sync03"
```

**サーバ用ステータスファイルの定義例**

```
set pd_sts_file_name_1="f1sts1","C:¥sysarea01¥f1sts01a"¥
                               ,"C:¥sysarea02¥f1sts01b"
set pd_sts_file_name_2="f1sts2","C:¥sysarea02¥f1sts02a"¥
                               ,"C:¥sysarea03¥f1sts02b"
set pd_sts_file_name_3="f1sts3","C:¥sysarea03¥f1sts03a"¥
                               ,"C:¥sysarea01¥f1sts03b"
```

### (c) BES1 のバックエンドサーバ定義

BES1 のバックエンドサーバ定義にシステムログファイル，シンクポイントダンプファイル，及びサーバ用ステータスファイルを定義します。

**システムログファイルの定義例**

```
set pd_log_rec_leng=4096
pdlogadfg -d sys -g b1log1 ONL
pdlogadfg -d sys -g b1log2 ONL
pdlogadfg -d sys -g b1log3 ONL
pdlogadpf -d sys -g b1log1 -a "C:¥sysarea01¥b1log01a"¥
                           -b "C:¥sysarea02¥b1log01b"
pdlogadpf -d sys -g b1log2 -a "C:¥sysarea02¥b1log02a"¥
                           -b "C:¥sysarea03¥b1log02b"
pdlogadpf -d sys -g b1log3 -a "C:¥sysarea03¥b1log03a"¥
                           -b "C:¥sysarea01¥b1log03b"
```

**シンクポイントダンプファイルの定義例**

```
pdlogadfg -d spd -g b1sync1 ONL
pdlogadfg -d spd -g b1sync2 ONL
pdlogadfg -d spd -g b1sync3 ONL
pdlogadpf -d spd -g b1sync1 -a "C:¥sysarea01¥b1sync01"
pdlogadpf -d spd -g b1sync2 -a "C:¥sysarea02¥b1sync02"
pdlogadpf -d spd -g b1sync3 -a "C:¥sysarea03¥b1sync03"
```

**サーバ用ステータスファイルの定義例**

```
set pd_sts_file_name_1="b1sts1","C:¥sysarea01¥b1sts01a"¥
                               ,"C:¥sysarea02¥b1sts01b"
set pd_sts_file_name_2="b1sts2","C:¥sysarea02¥b1sts02a"¥
                               ,"C:¥sysarea03¥b1sts02b"
set pd_sts_file_name_3="b1sts3","C:¥sysarea03¥b1sts03a"¥
                               ,"C:¥sysarea01¥b1sts03b"
```

### (d) BES2 のバックエンドサーバ定義

BES2 のバックエンドサーバ定義にシステムログファイル，シンクポイントダンプファイル，及びサーバ用ステータスファイルを定義します。

**システムログファイルの定義例**

```
set pd_log_rec_leng=4096
pdlogadfg -d sys -g b2log1 ONL
pdlogadfg -d sys -g b2log2 ONL
pdlogadfg -d sys -g b2log3 ONL
pdlogadpf -d sys -g b2log1 -a "C:¥sysarea01¥b2log01a"¥
                           -b "C:¥sysarea02¥b2log01b"
pdlogadpf -d sys -g b2log2 -a "C:¥sysarea02¥b2log02a"¥
                           -b "C:¥sysarea03¥b2log02b"
pdlogadpf -d sys -g b2log3 -a "C:¥sysarea03¥b2log03a"¥
                           -b "C:¥sysarea01¥b2log03b"
```

**シンクポイントダンプファイルの定義例**

```
pdlogadfg -d spd -g b2sync1 ONL
pdlogadfg -d spd -g b2sync2 ONL
pdlogadfg -d spd -g b2sync3 ONL
pdlogadpf -d spd -g b2sync1 -a "C:¥sysarea01¥b2sync01"
pdlogadpf -d spd -g b2sync2 -a "C:¥sysarea02¥b2sync02"
pdlogadpf -d spd -g b2sync3 -a "C:¥sysarea03¥b2sync03"
```

**サーバ用ステータスファイルの定義例**

```
set pd_sts_file_name_1="b2sts1","C:¥sysarea01¥b2sts01a"¥
                               ,"C:¥sysarea02¥b2sts01b"
set pd_sts_file_name_2="b2sts2","C:¥sysarea02¥b2sts02a"¥
                               ,"C:¥sysarea03¥b2sts02b"
set pd_sts_file_name_3="b2sts3","C:¥sysarea03¥b2sts03a"¥
                               ,"C:¥sysarea01¥b2sts03b"
```

## (2) HiRDB ファイルシステム領域の作成

pdfmkfs コマンドで HiRDB ファイルシステム領域を作成します。

**コマンドの入力例**

```
pdfmkfs -n 50 -l 20 -i -k SYS C:¥sysarea01
pdfmkfs -n 50 -l 20 -i -k SYS C:¥sysarea02
pdfmkfs -n 50 -l 20 -i -k SYS C:¥sysarea03
```

## (3) システムファイルの作成

### (a) システムログファイルの作成

pdloginit コマンドでシステムログファイルを作成します。

**コマンドの入力例（FES1 用）**

```
pdloginit -d sys -s f001 -f C:¥sysarea01¥f1log01a -n 1024
pdloginit -d sys -s f001 -f C:¥sysarea01¥f1log03b -n 1024
pdloginit -d sys -s f001 -f C:¥sysarea02¥f1log02a -n 1024
pdloginit -d sys -s f001 -f C:¥sysarea02¥f1log01b -n 1024
pdloginit -d sys -s f001 -f C:¥sysarea03¥f1log03a -n 1024
pdloginit -d sys -s f001 -f C:¥sysarea03¥f1log02b -n 1024
```

**コマンドの入力例（BES1 用）**

```
pdloginit -d sys -s b001 -f C:¥sysarea01¥b1log01a -n 1024
pdloginit -d sys -s b001 -f C:¥sysarea01¥b1log03b -n 1024
pdloginit -d sys -s b001 -f C:¥sysarea02¥b1log02a -n 1024
pdloginit -d sys -s b001 -f C:¥sysarea02¥b1log01b -n 1024
pdloginit -d sys -s b001 -f C:¥sysarea03¥b1log03a -n 1024
pdloginit -d sys -s b001 -f C:¥sysarea03¥b1log02b -n 1024
```

**コマンドの入力例（BES2 用）**

```
pdloginit -d sys -s b002 -f C:¥sysarea01¥b2log01a -n 1024
pdloginit -d sys -s b002 -f C:¥sysarea01¥b2log03b -n 1024
pdloginit -d sys -s b002 -f C:¥sysarea02¥b2log02a -n 1024
pdloginit -d sys -s b002 -f C:¥sysarea02¥b2log01b -n 1024
pdloginit -d sys -s b002 -f C:¥sysarea03¥b2log03a -n 1024
pdloginit -d sys -s b002 -f C:¥¥syarea03¥b2log02b -n 1024
```

### (b) シンクポイントダンプファイルの作成

pdloginit コマンドでシンクポイントダンプファイルを作成します。

**コマンドの入力例（FES1 用）**

```
pdloginit -d spd -s f001 -f C:¥sysarea01¥f1sync01 -n 64
pdloginit -d spd -s f001 -f C:¥sysarea02¥f1sync02 -n 64
pdloginit -d spd -s f001 -f C:¥sysarea03¥f1sync03 -n 64
```

**コマンドの入力例（BES1 用）**

```
pdloginit -d spd -s b001 -f C:¥sysarea01¥b1sync01 -n 64
pdloginit -d spd -s b001 -f C:¥sysarea02¥b1sync02 -n 64
pdloginit -d spd -s b001 -f C:¥sysarea03¥b1sync03 -n 64
```

**コマンドの入力例（BES2 用）**

```
pdloginit -d spd -s b002 -f C:¥sysarea01¥b2sync01 -n 64
pdloginit -d spd -s b002 -f C:¥sysarea02¥b2sync02 -n 64
pdloginit -d spd -s b002 -f C:¥sysarea03¥b2sync03 -n 64
```

**（c）サーバ用ステータスファイルの作成**

pdstsinit コマンドでサーバ用ステータスファイルを作成します。

**コマンドの入力例（FES1 用）**

```
pdstsinit -s f001 -f C:¥sysarea01¥f1sts01a -l 4096 -c 256
pdstsinit -s f001 -f C:¥sysarea01¥f1sts03b -l 4096 -c 256
pdstsinit -s f001 -f C:¥sysarea02¥f1sts02a -l 4096 -c 256
pdstsinit -s f001 -f C:¥sysarea02¥f1sts01b -l 4096 -c 256
pdstsinit -s f001 -f C:¥sysarea03¥f1sts03a -l 4096 -c 256
pdstsinit -s f001 -f C:¥sysarea03¥f1sts02b -l 4096 -c 256
```

**コマンドの入力例（BES1 用）**

```
pdstsinit -s b001 -f C:¥sysarea01¥b1sts01a -l 4096 -c 256
pdstsinit -s b001 -f C:¥sysarea01¥b1sts03b -l 4096 -c 256
pdstsinit -s b001 -f C:¥sysarea02¥b1sts02a -l 4096 -c 256
pdstsinit -s b001 -f C:¥sysarea02¥b1sts01b -l 4096 -c 256
pdstsinit -s b001 -f C:¥sysarea03¥b1sts03a -l 4096 -c 256
pdstsinit -s b001 -f C:¥sysarea03¥b1sts02b -l 4096 -c 256
```

**コマンドの入力例（BES2 用）**

```
pdstsinit -s b002 -f C:¥sysarea01¥b2sts01a -l 4096 -c 256
pdstsinit -s b002 -f C:¥sysarea01¥b2sts03b -l 4096 -c 256
pdstsinit -s b002 -f C:¥sysarea02¥b2sts02a -l 4096 -c 256
pdstsinit -s b002 -f C:¥sysarea02¥b2sts01b -l 4096 -c 256
pdstsinit -s b002 -f C:¥sysarea03¥b2sts03a -l 4096 -c 256
pdstsinit -s b002 -f C:¥sysarea03¥b2sts02b -l 4096 -c 256
```

**（d）ユニット用ステータスファイルの作成**

pdstsinit コマンドでユニット用ステータスファイルを作成します。

**コマンドの入力例**

```
pdstsinit -u unt1 -f C:¥sysarea01¥u1sts01a -l 4096 -c 256
pdstsinit -u unt1 -f C:¥sysarea01¥u1sts03b -l 4096 -c 256
pdstsinit -u unt1 -f C:¥sysarea02¥u1sts02a -l 4096 -c 256
pdstsinit -u unt1 -f C:¥sysarea02¥u1sts01b -l 4096 -c 256
pdstsinit -u unt1 -f C:¥sysarea03¥u1sts03a -l 4096 -c 256
pdstsinit -u unt1 -f C:¥sysarea03¥u1sts02b -l 4096 -c 256
```

# 4.5 システム用 RD エリアの作成

**実行者** HiRDB 管理者

HiRDB を初期開始するときは，**データベース初期設定ユティリティ（pdinit）**でシステム用 RD エリアを作成する必要があります。

データベース初期設定ユティリティ（pdinit）は，HiRDB を初期開始する場合（インストールしてから最初に pdstart コマンドを実行するとき）に，コマンドの入力要求が来たときに実行します。それ以外のタイミングでデータベース初期設定ユティリティ（pdinit）は実行できません。

ここでは，データベース初期設定ユティリティ（pdinit）の引数として指定する**制御文ファイル**の内容とデータベース初期設定ユティリティ（pdinit）の実行例について説明します。システム用 RD エリアは，create rdarea 文で作成します。

システム用 RD エリアとは，次に示す RD エリアのことです。

- マスタディレクトリ用 RD エリア
- データディレクトリ用 RD エリア
- データディクショナリ用 RD エリア

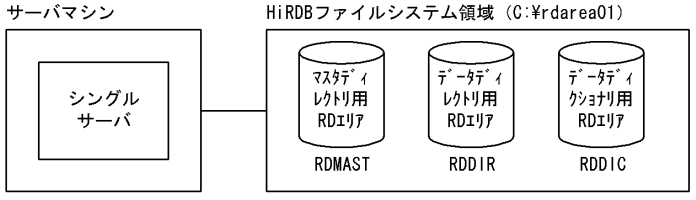
## 4.5.1 基本事項

1. システム用 RD エリアは，「4.3　HiRDB ファイルシステム領域の作成」で作成した RD エリア用の HiRDB ファイルシステム領域に作成します。
2. HiRDB/ パラレルサーバの場合，ディクショナリサーバを定義したサーバマシンの HiRDB ファイルシステム領域にシステム用 RD エリアを作成します。
3. システム用 RD エリアの設計方法については，HiRDB/ シングルサーバの場合は「8.4　RD エリアの配置」を，HiRDB/ パラレルサーバの場合は「9.4　RD エリアの配置」を参照してください。
4. HiRDB/ パラレルサーバの場合，システムマネジャを定義したサーバマシンでデータベース初期設定ユティリティ（pdinit）を実行してください。
5. ここの説明では，データベース初期設定ユティリティ（pdinit）の create rdarea 文でシステム用 RD エリアしか作成しません。これはシステム用 RD エリアが HiRDB の稼働に必要であるためです。ただし，次に示す RD エリアもデータベース初期設定ユティリティ（pdinit）の create rdarea 文で定義できます。
   - ユーザ用 RD エリア
   - ユーザ LOB 用 RD エリア
   - データディクショナリ LOB 用 RD エリア
   - リスト用 RD エリア

## 4.5.2 例題 1（HiRDB/ シングルサーバの場合）

次に示す RD エリア用の HiRDB ファイルシステム領域に，システム用 RD エリアを作成します。

- C:¥rdarea01




## （1）制御文ファイルの作成

データベース初期設定ユティリティ（pdinit）の引数に指定する制御文ファイルを作成します。制御文ファイルを作成する場所は任意ですが，ここでは次に示すファイル名で作成することにします。

- C:¥hirdb¥pdinit01

**制御文ファイルの内容**

```
create rdarea RDMAST for masterdirectory                    1
    file name "C:¥rdarea01¥rdmast01"
          initial 10 segments;
create rdarea RDDIR for datadirectory                       2
    file name "C:¥rdarea01¥rddir01"
          initial 5 segments;
create rdarea RDDIC for datadictionary                      3
    extension use 50 segments
    file name "C:¥rdarea01¥rddic01"
          initial 20 segments;
```

〔説明〕

1. マスタディレクトリ用 RD エリアの定義
   HiRDB ファイルシステム領域に rdmast01 という HiRDB ファイルを作成します。HiRDB ファイル内のセグメント数を 10 とします。
2. データディレクトリ用 RD エリアの定義
   HiRDB ファイルシステム領域に rddir01 という HiRDB ファイルを作成します。HiRDB ファイル内のセグメント数を 5 とします。
3. データディクショナリ用 RD エリアの定義
   HiRDB ファイルシステム領域に rddic01 という HiRDB ファイルを作成します。HiRDB ファイル内のセグメント数を 20 とします。
   RD エリアの自動増分機能を使用します。増分量を 50 セグメントとします。

## （2）データベース初期設定ユティリティ（pdinit）の実行

**コマンドの入力例**

```
pdinit -d C:¥hirdb¥pdinit01
```

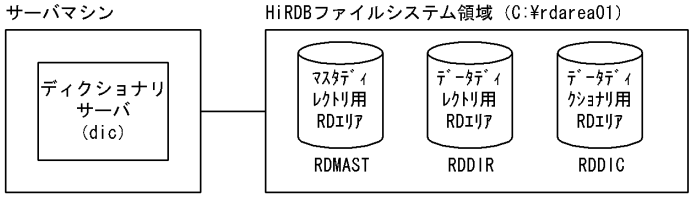
〔説明〕

**-d：**（1）で作成した制御文ファイルのファイル名を指定します。

## 4.5.3 例題 2（HiRDB/ パラレルサーバの場合）

次に示す RD エリア用の HiRDB ファイルシステム領域に，システム用 RD エリアを作成します。システム用 RD エリアは，ディクショナリサーバを定義したサーバマシンに作成します。

- C:¥rdarea01




### (1) 制御文ファイルの作成

データベース初期設定ユティリティ（pdinit）の引数に指定する制御文ファイルを作成します。制御文ファイルを作成する場所は任意ですが，ここでは次に示すファイル名で作成することにします。

- C:¥hirdb¥pdinit01

**制御文ファイルの内容**

```
create rdarea RDMAST for masterdirectory                    1
    server name dic
    file name "C:¥rdarea01¥rdmast01"
         initial 10 segments;
create rdarea RDDIR for datadirectory                       2
    server name dic
    file name "C:¥rdarea01¥rddir01"
         initial 5 segments;
create rdarea RDDIC for datadictionary                      3
    server name dic
    extension use 50 segments
    file name "C:¥rdarea01¥rddic01"
         initial 20 segments;
```

〔説明〕

1. マスタディレクトリ用 RD エリアの定義
   マスタディレクトリ用 RD エリアを管理するディクショナリサーバの名称（dic）を指定します。HiRDB ファイルシステム領域に rdmast01 という HiRDB ファイルを作成します。HiRDB ファイル内のセグメント数を 10 とします。
2. データディレクトリ用 RD エリアの定義
   データディレクトリ用 RD エリアを管理するディクショナリサーバの名称（dic）を指定します。HiRDB ファイルシステム領域に rddir01 という HiRDB ファイルを作成します。HiRDB ファイル内のセグメント数を 5 とします。
3. データディクショナリ用 RD エリアの定義
   データディクショナリ用 RD エリアを管理するディクショナリサーバの名称（dic）を指定します。HiRDB ファイルシステム領域に rddic01 という HiRDB ファイルを作成します。HiRDB ファイル内のセグメント数を 20 とします。
   RD エリアの自動増分機能を使用します。増分量を 50 セグメントとします。

## (2) データベース初期設定ユティリティ（pdinit）の実行

**コマンドの入力例**

```
pdinit -d C:¥hirdb¥pdinit01
```

〔説明〕

-d：(1) で作成した制御文ファイルのファイル名を指定します。

# 4.6 HiRDB の初期開始

**実行者** HiRDB 管理者

「4.5 システム用 RD エリアの作成」で説明したデータベース初期設定ユティリティ（pdinit）は，HiRDB の初期開始コマンド（pdstart コマンド）を実行してから，その実行途中でだけ実行できます。

## （1） HiRDB の初期開始の方法

HiRDB ファイルシステム領域を作成してから HiRDB を最初に開始（初期開始）するときは，pdstart コマンドを実行します。初期開始のために pdstart コマンドを実行すると，データベース初期設定ユティリティ（pdinit）の実行を要求するメッセージが表示されます。

- HiRDB/ シングルサーバを開始する場合は，シングルサーバを定義したサーバマシンから pdstart コマンドを実行してください。
- HiRDB/ パラレルサーバを開始する場合は，システムマネジャを定義したサーバマシンから pdstart コマンドを実行してください。

## （2） RD エリアの作成の前提条件

次に示す RD エリアを作成するときは，HiRDB が稼働中であることが前提です。事前に HiRDB を開始しておいてください。

- ユーザ用 RD エリア
- ユーザ LOB 用 RD エリア
- データディクショナリ LOB 用 RD エリア
- リスト用 RD エリア

# 4.7 ユーザ用 RD エリアの作成

**実行者** HiRDB 管理者

表及びインデクスを格納するユーザ用 RD エリアを作成します。ユーザ用 RD エリアは，**データベース構成変更ユティリティ（**pdmod**）**の create rdarea 文で作成します。

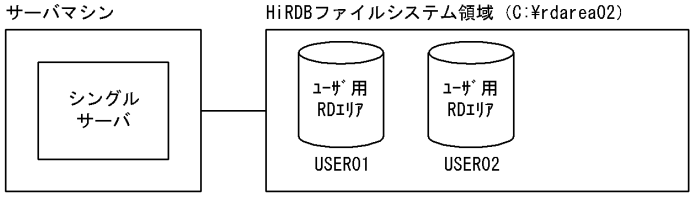
## 4.7.1 基本事項

1.「4.3 HiRDB ファイルシステム領域の作成」で作成した RD エリア用の HiRDB ファイルシステム領域に，ユーザ用 RD エリアを作成します。
2. HiRDB/ パラレルサーバの場合，バックエンドサーバを定義したサーバマシンの HiRDB ファイルシステム領域にユーザ用 RD エリアを作成します。
3. ユーザ用 RD エリアの設計方法については，HiRDB/ シングルサーバの場合は「8.4 RD エリアの配置」を，HiRDB/ パラレルサーバの場合は「9.4 RD エリアの配置」を参照してください。
4. HiRDB/ パラレルサーバの場合，システムマネジャを定義したサーバマシンでデータベース構成変更ユティリティ（pdmod）を実行してください。
5. ユーザ用 RD エリアを作成する前に，HiRDB が稼働しているかどうかを pdls コマンドで確認してください。HiRDB/ パラレルサーバの場合は，システムマネジャを定義したサーバマシンから pdls コマンドを入力してください。
6. HiRDB が稼働していない場合，pdstart コマンドで HiRDB を開始してください。HiRDB/ パラレルサーバを開始する場合は，システムマネジャを定義したサーバマシンから pdstart コマンドを入力してください。

## 4.7.2 例題 1（HiRDB/ シングルサーバの場合）

次に示す RD エリア用の HiRDB ファイルシステム領域に，ユーザ用 RD エリアを作成します。

- C:¥rdarea02




### (1) 制御文ファイルの作成

データベース構成変更ユティリティ（pdmod）の引数に指定する制御文ファイルを作成します。制御文ファイルを作成する場所は任意ですが，ここでは次に示すファイル名で作成することにします。

- C:¥hirdb¥pdmod01

**制御文ファイルの内容**

```
create rdarea USER01 for user used by PUBLIC                    1
    extension use 50 segments
    file name "C:¥rdarea02¥user01"
         initial 500 segments;
create rdarea USER02 for user used by PUBLIC                    2
    extension use 50 segments
    file name "C:¥rdarea02¥user02"
         initial 500 segments;
```

〔説明〕

1. ユーザ用 RD エリア（USER01）の定義
   USER01 を公用 RD エリア（PUBLIC）とします。HiRDB ファイルシステム領域に user01 という HiRDB ファイルを作成します。HiRDB ファイル内のセグメント数を 500 とします。
   RD エリアの自動増分機能を使用します。増分量を 50 セグメントとします。
2. ユーザ用 RD エリア（USER02）の定義
   USER02 を公用 RD エリア（PUBLIC）とします。HiRDB ファイルシステム領域に user02 という HiRDB ファイルを作成します。HiRDB ファイル内のセグメント数を 500 とします。
   RD エリアの自動増分機能を使用します。増分量を 50 セグメントとします。

### (2) データベース構成変更ユティリティ（pdmod）の実行

**コマンドの入力例**

```
pdmod -a C:¥hirdb¥pdmod01
```

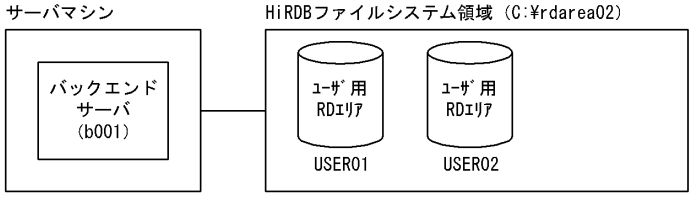
〔説明〕

-a：(1) で作成した制御文ファイルのファイル名を指定します。

## 4.7.3 例題 2（HiRDB/ パラレルサーバの場合）

次に示す RD エリア用の HiRDB ファイルシステム領域に，ユーザ用 RD エリアを作成します。ユーザ用 RD エリアは，バックエンドサーバを定義したサーバマシンに作成します。

- C:¥rdarea02




### (1) 制御文ファイルの作成

データベース構成変更ユティリティ（pdmod）の引数に指定する制御文ファイルを作成します。制御文ファイルを作成する場所は任意ですが，ここでは次に示すファイル名で作成することにします。

- C:¥hirdb¥pdmod01

**制御文ファイルの内容**

```
create rdarea USER01 for user used by PUBLIC                  1
    server name b001
    extension use 50 segments
    file name "C:¥rdarea02¥user01"
         initial 500 segments;
create rdarea USER02 for user used by PUBLIC                  2
    server name b001
    extension use 50 segments
    file name "C:¥rdarea02¥user02"
         initial 500 segments;
```

〔説明〕

1. ユーザ用 RD エリア（USER01）の定義
   USER01 を公用 RD エリア（PUBLIC）とします。USER01 を管理するバックエンドサーバの名称（b001）を指定します。HiRDB ファイルシステム領域に user01 という HiRDB ファイルを作成します。HiRDB ファイル内のセグメント数を 500 とします。
   RD エリアの自動増分機能を使用します。増分量を 50 セグメントとします。
2. ユーザ用 RD エリア（USER02）の定義
   USER02 を公用 RD エリア（PUBLIC）とします。USER02 を管理するバックエンドサーバの名称（b001）を指定します。HiRDB ファイルシステム領域に user02 という HiRDB ファイルを作成します。HiRDB ファイル内のセグメント数を 500 とします。
   RD エリアの自動増分機能を使用します。増分量を 50 セグメントとします。

## （2）データベース構成変更ユティリティ（pdmod）の実行

**コマンドの入力例**

```
pdmod -a C:¥hirdb¥pdmod01
```

〔説明〕

-a：（1）で作成した制御文ファイルのファイル名を指定します。

# 4.8 ユーザ LOB 用 RD エリアの作成

**実行者** HiRDB 管理者

LOB 属性のデータを使用する場合，LOB 属性のデータを格納するユーザ LOB 用 RD エリアが必要になります。ユーザ LOB 用 RD エリアは，**データベース構成変更ユティリティ（**pdmod**）**の create rdarea 文で作成します。

## 4.8.1 基本事項

1.「4.3　HiRDB ファイルシステム領域の作成」で作成した RD エリア用の HiRDB ファイルシステム領域に，ユーザ LOB 用 RD エリアを作成します。
2. HiRDB/ パラレルサーバの場合，バックエンドサーバを定義したサーバマシンの HiRDB ファイルシステム領域にユーザ LOB 用 RD エリアを作成します。
3. ユーザ LOB 用 RD エリアの設計方法については，HiRDB/ シングルサーバの場合は「8.4　RD エリアの配置」を，HiRDB/ パラレルサーバの場合は「9.4　RD エリアの配置」を参照してください。
4. HiRDB/ パラレルサーバの場合，システムマネジャを定義したサーバマシンでデータベース構成変更ユティリティ（pdmod）を実行してください。
5. ユーザ LOB 用 RD エリアを作成する前に，HiRDB が稼働しているかどうかを pdls コマンドで確認してください。HiRDB/ パラレルサーバの場合は，システムマネジャを定義したサーバマシンから pdls コマンドを入力してください。
6. HiRDB が稼働していない場合，pdstart コマンドで HiRDB を開始してください。HiRDB/ パラレルサーバを開始する場合は，システムマネジャを定義したサーバマシンから pdstart コマンドを入力してください。

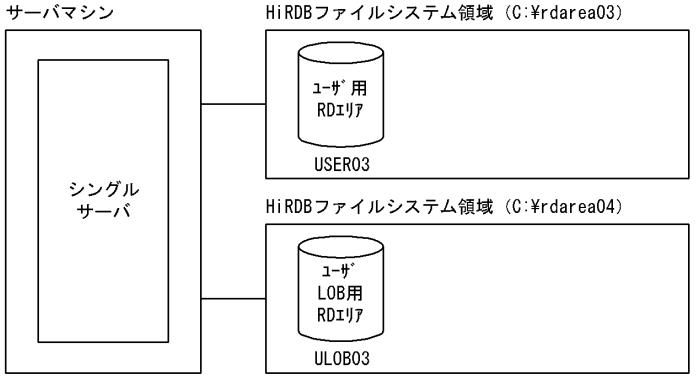
## 4.8.2 例題 1（HiRDB/ シングルサーバの場合）

次に示す RD エリア用の HiRDB ファイルシステム領域に，ユーザ用 RD エリアを作成します。このユーザ用 RD エリアには，LOB 列構成基表を格納します。

- C:¥rdarea03

さらに，次に示す RD エリア用の HiRDB ファイルシステム領域にユーザ LOB 用 RD エリアを作成します。このユーザ LOB 用 RD エリアには，LOB 属性のデータを格納します。

- C:¥rdarea04




## （1）制御文ファイルの作成

データベース構成変更ユティリティ（pdmod）の引数に指定する制御文ファイルを作成します。制御文ファイルを作成する場所は任意ですが，ここでは次に示すファイル名で作成することにします。

- C:¥hirdb¥pdmod02

**制御文ファイルの内容**

```
create rdarea USER03 for user used by PUBLIC                1
    extension use 50 segments
    file name "C:¥rdarea03¥user03"
         initial 500 segments;
create rdarea ULOB03 for LOB used by PUBLIC                 2
    extension use 50 segments
    file name "C:¥rdarea04¥ulob03"
         initial 20000 segments;
```

〔説明〕

1. ユーザ用 RD エリア（USER03）の定義
   USER03 を公用 RD エリア（PUBLIC）とします。HiRDB ファイルシステム領域に user03 という HiRDB ファイルを作成します。HiRDB ファイル内のセグメント数を 500 とします。
   RD エリアの自動増分機能を使用します。増分量を 50 セグメントとします。
2. ユーザ LOB 用 RD エリア（ULOB03）の定義
   ULOB03 を公用 RD エリア（PUBLIC）とします。HiRDB ファイルシステム領域に ulob03 という HiRDB ファイルを作成します。HiRDB ファイル内のセグメント数を 20000 とします。
   RD エリアの自動増分機能を使用します。増分量を 50 セグメントとします。

## （2）データベース構成変更ユティリティ（pdmod）の実行

**コマンドの入力例**

```
pdmod -a C:¥hirdb¥pdmod02
```

〔説明〕

-a：（1）で作成した制御文ファイルのファイル名を指定します。

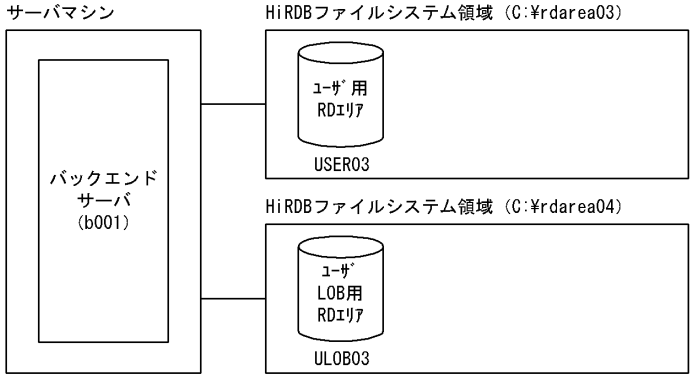
## 4.8.3 例題 2（HiRDB/ パラレルサーバの場合）

次に示す RD エリア用の HiRDB ファイルシステム領域にユーザ用 RD エリアを作成します。このユーザ用 RD エリアには，LOB 列構成基表を格納します。

- C:¥rdarea03

さらに，次に示す RD エリア用の HiRDB ファイルシステム領域にユーザ LOB 用 RD エリアを作成します。このユーザ LOB 用 RD エリアには，LOB 属性のデータを格納します。

- C:¥rdarea04




### (1) 制御文ファイルの作成

データベース構成変更ユティリティ（pdmod）の引数に指定する制御文ファイルを作成します。制御文ファイルを作成する場所は任意ですが，ここでは次に示すファイル名で作成することにします。

- C:¥hirdb¥pdmod02

**制御文ファイルの内容**

```
create rdarea USER03 for user used by PUBLIC                1
    server name b001
    extension use 50 segments
    file name "C:¥rdarea03¥user03"
         initial 500 segments;
create rdarea ULOB03 for LOB used by PUBLIC                 2
    server name b001
    extension use 50 segments
    file name "C:¥rdarea04¥ulob03"
         initial 20000 segments;
```

〔説明〕

1. ユーザ用 RD エリア（USER03）の定義
   USER03 を公用 RD エリア（PUBLIC）とします。USER03 を管理するバックエンドサーバの名称（b001）を指定します。HiRDB ファイルシステム領域に user03 という HiRDB ファイルを作成します。HiRDB ファイル内のセグメント数を 500 とします。
   RD エリアの自動増分機能を使用します。増分量を 50 セグメントとします。
2. ユーザ LOB 用 RD エリア（ULOB03）の定義です。
   ULOB03 を公用 RD エリア（PUBLIC）とします。ULOB03 を管理するバックエンドサーバの名

称（b001）を指定します。HiRDB ファイルシステム領域に ulob03 という HiRDB ファイルを作成します。HiRDB ファイル内のセグメント数を 20000 とします。
RD エリアの自動増分機能を使用します。増分量を 50 セグメントとします。

## （2） データベース構成変更ユティリティ（pdmod）の実行

**コマンドの入力例**

```
pdmod -a C:¥hirdb¥pdmod02
```

〔説明〕

-a：（1）で作成した制御文ファイルのファイル名を指定します。

# 4.9 データディクショナリ LOB 用 RD エリアの作成

**実行者** HiRDB 管理者

ストアドプロシジャ又はストアドファンクションを使用する場合，データディクショナリ LOB 用 RD エリアが必要になります。データディクショナリ LOB 用 RD エリアは，**データベース構成変更ユティリティ**（pdmod）の create rdarea 文で作成します。

データディクショナリ LOB 用 RD エリアは，次に示す用途ごとに用意する必要があります。

- ストアドプロシジャ又はストアドファンクションの定義ソース格納用のデータディクショナリ LOB 用 RD エリア
- ストアドプロシジャ又はストアドファンクションの SQL オブジェクト格納用のデータディクショナリ LOB 用 RD エリア

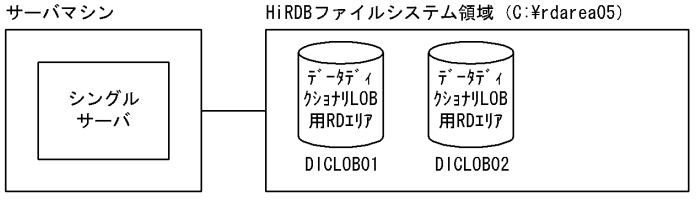
## 4.9.1 基本事項

1.「4.3 HiRDB ファイルシステム領域の作成」で作成した RD エリア用の HiRDB ファイルシステム領域に，データディクショナリ LOB 用 RD エリアを作成します。
2. HiRDB/ パラレルサーバの場合，ディクショナリサーバを定義したサーバマシンの HiRDB ファイルシステム領域にデータディクショナリ LOB 用 RD エリアを作成します。
3. データディクショナリ LOB 用 RD エリアの設計方法については，HiRDB/ シングルサーバの場合は「8.4 RD エリアの配置」を，HiRDB/ パラレルサーバの場合は「9.4 RD エリアの配置」を参照してください。
4. HiRDB/ パラレルサーバの場合，システムマネジャを定義したサーバマシンでデータベース構成変更ユティリティ（pdmod）を実行してください。
5. データディクショナリ LOB 用 RD エリアを作成する前に，HiRDB が稼働しているかどうかを pdls コマンドで確認してください。HiRDB/ パラレルサーバの場合は，システムマネジャを定義したサーバマシンから pdls コマンドを入力してください。
6. HiRDB が稼働していない場合，pdstart コマンドで HiRDB を開始してください。HiRDB/ パラレルサーバを開始する場合は，システムマネジャを定義したサーバマシンから pdstart コマンドを入力してください。

## 4.9.2 例題 1（HiRDB/ シングルサーバの場合）

次に示す RD エリア用の HiRDB ファイルシステム領域に，データディクショナリ LOB 用 RD エリアを作成します。

- C:¥rdarea05



### (1) 制御文ファイルの作成

データベース構成変更ユティリティ（pdmod）の引数に指定する制御文ファイルを作成します。制御文ファイルを作成する場所は任意ですが，ここでは次に示すファイル名で作成することにします。

- C:¥hirdb¥pdmod03

**制御文ファイルの内容**

```
create rdarea DICLOB01 for LOB used by HiRDB(SQL_ROUTINES)    1
    extension use 1000 segments
    file name "C:¥rdarea05¥diclob01"
        initial 10000 segments;
create rdarea DICLOB02 for LOB used by HiRDB(SQL_ROUTINES)    2
    extension use 1000 segments
    file name "C:¥rdarea05¥diclob02"
        initial 10000 segments;
```

〔説明〕

1. データディクショナリ LOB 用 RD エリア（DICLOB01）の定義
   DICLOB01 は，定義ソース格納用のデータディクショナリ LOB 用 RD エリアになります。最初に定義したデータディクショナリ LOB 用 RD エリアが定義ソース格納用になります。HiRDB ファイルシステム領域に diclob01 という HiRDB ファイルを作成します。HiRDB ファイル内のセグメント数を 10000 とします。
   RD エリアの自動増分機能を使用します。増分量を 1000 セグメントとします。
2. データディクショナリ LOB 用 RD エリア（DICLOB02）の定義
   DICLOB02 は，SQL オブジェクト格納用のデータディクショナリ LOB 用 RD エリアになります。2 番目に定義したデータディクショナリ LOB 用 RD エリアが SQL オブジェクト格納用になります。HiRDB ファイルシステム領域に diclob02 という HiRDB ファイルを作成します。HiRDB ファイル内のセグメント数を 10000 とします。
   RD エリアの自動増分機能を使用します。増分量を 1000 セグメントとします。

### (2) データベース構成変更ユティリティ（pdmod）の実行

**コマンドの入力例**

```
pdmod -a C:¥hirdb¥pdmod03
```

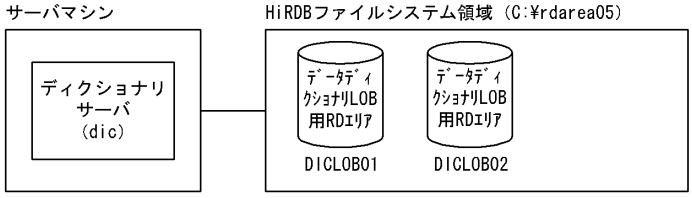
〔説明〕

-a：(1) で作成した制御文ファイルのファイル名を指定します。

## 4.9.3 例題 2（HiRDB/ パラレルサーバの場合）

次に示す RD エリア用の HiRDB ファイルシステム領域に，データディクショナリ LOB 用 RD エリアを作成します。

- C:¥rdarea05




## (1) 制御文ファイルの作成

データベース構成変更ユティリティ（pdmod）の引数に指定する制御文ファイルを作成します。制御文ファイルを作成する場所は任意ですが，ここでは次に示すファイル名で作成することにします。

- C:¥hirdb¥pdmod03

**制御文ファイルの内容**

```
create rdarea DICLOB01 for LOB used by HiRDB(SQL_ROUTINES)   1
    server name dic
    extension use 1000 segments
    file name "C:¥rdarea05¥diclob01"
        initial 10000 segments;
create rdarea DICLOB02 for LOB used by HiRDB(SQL_ROUTINES)   2
    server name dic
    extension use 1000 segments
    file name "C:¥rdarea05¥diclob02"
        initial 10000 segments;
```

〔説明〕

1. データディクショナリ LOB 用 RD エリア（DICLOB01）の定義
   DICLOB01 は，定義ソース格納用のデータディクショナリ LOB 用 RD エリアになります。最初に定義したデータディクショナリ LOB 用 RD エリアが定義ソース格納用になります。データディクショナリ LOB 用 RD エリアを管理するディクショナリサーバの名称（dic）を指定します。
   HiRDB ファイルシステム領域に diclob01 という HiRDB ファイルを作成します。HiRDB ファイル内のセグメント数を 10000 とします。
   RD エリアの自動増分機能を使用します。増分量を 1000 セグメントとします。
2. データディクショナリ LOB 用 RD エリア（DICLOB02）の定義
   DICLOB02 は，SQL オブジェクト格納用のデータディクショナリ LOB 用 RD エリアになります。2 番目に定義したデータディクショナリ LOB 用 RD エリアが SQL オブジェクト格納用になります。データディクショナリ LOB 用 RD エリアを管理するディクショナリサーバの名称（dic）を指定します。HiRDB ファイルシステム領域に diclob02 という HiRDB ファイルを作成します。
   HiRDB ファイル内のセグメント数を 10000 とします。
   RD エリアの自動増分機能を使用します。増分量を 1000 セグメントとします。

## (2) データベース構成変更ユティリティ（pdmod）の実行

**コマンドの入力例**

```
pdmod -a C:¥hirdb¥pdmod03
```

〔説明〕

-a：（1）で作成した制御文ファイルのファイル名を指定します。

# 4.10 リスト用 RD エリアの作成

**実行者** HiRDB 管理者

絞込み検索をする場合は，リスト用 RD エリアが必要になります。リスト用 RD エリアは，**データベース構成変更ユティリティ（**pdmod**）**の create rdarea 文で作成します。

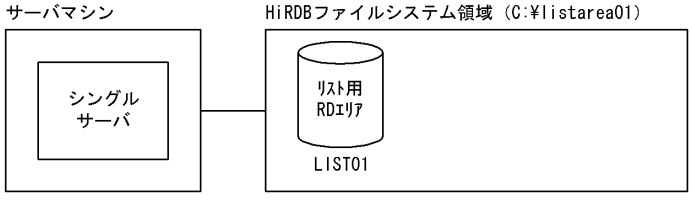
## 4.10.1 基本事項

1.「4.3 HiRDB ファイルシステム領域の作成」で作成したリスト用 RD エリア用の HiRDB ファイルシステム領域に，リスト用 RD エリアを作成します。
2. HiRDB/ パラレルサーバの場合，バックエンドサーバ（基表があるバックエンドサーバ）を定義したサーバマシンの HiRDB ファイルシステム領域にリスト用 RD エリアを作成します。
3. リスト用 RD エリアの設計方法については，HiRDB/ シングルサーバの場合は「8.4 RD エリアの配置」を，HiRDB/ パラレルサーバの場合は「9.4 RD エリアの配置」を参照してください。
4. HiRDB/ パラレルサーバの場合，システムマネジャを定義したサーバマシンでデータベース構成変更ユティリティ（pdmod）を実行してください。
5. リスト用 RD エリアを作成する前に，HiRDB が稼働しているかどうかを pdls コマンドで確認してください。HiRDB/ パラレルサーバの場合は，システムマネジャを定義したサーバマシンから pdls コマンドを入力してください。
6. HiRDB が稼働していない場合，pdstart コマンドで HiRDB を開始してください。HiRDB/ パラレルサーバを開始する場合は，システムマネジャを定義したサーバマシンから pdstart コマンドを入力してください。

## 4.10.2 例題 1（HiRDB/ シングルサーバの場合）

次に示すリスト用 RD エリア用の HiRDB ファイルシステム領域に，リスト用 RD エリアを作成します。

- C:¥listarea01




### (1) 制御文ファイルの作成

データベース構成変更ユティリティ（pdmod）の引数に指定する制御文ファイルを作成します。制御文ファイルを作成する場所は任意ですが，ここでは次に示すファイル名で作成することにします。

- C:¥hirdb¥pdmod04

**制御文ファイルの内容**

```
create rdarea LIST01 for list
    page 4096 characters storage control segment 2 pages      1
    file name "C:¥listarea01¥list01"
        initial 1000 segments;
```

〔説明〕

1. リスト用 RD エリア（LIST01）の定義
   RD エリアのページ長及びセグメントサイズを指定します。HiRDB ファイルシステム領域に list01 という HiRDB ファイルを作成します。HiRDB ファイル内のセグメント数を 1000 とします。

### (2) データベース構成変更ユティリティ（pdmod）の実行

**コマンドの入力例**

```
pdmod -a C:¥hirdb¥pdmod04
```

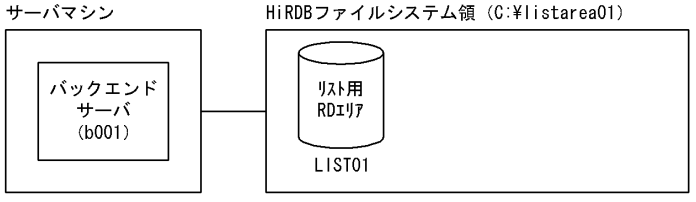
〔説明〕

-a：(1) で作成した制御文ファイルのファイル名を指定します。

## 4.10.3 例題 2（HiRDB/ パラレルサーバの場合）

次に示すリスト用 RD エリア用の HiRDB ファイルシステム領域に，リスト用 RD エリアを作成します。

- C:¥listarea01




### (1) 制御文ファイルの作成

データベース構成変更ユティリティ（pdmod）の引数に指定する制御文ファイルを作成します。制御文ファイルを作成する場所は任意ですが，ここでは次に示すファイル名で作成することにします。

- C:¥hirdb¥pdmod04

**制御文ファイルの内容**

```
create rdarea LIST01 for list                                  1
    server name b001
    page 4096 characters storage control segment 2 pages
    file name "C:¥listarea01¥list01"
        initial 1000 segments;
```

〔説明〕

1. リスト用 RD エリア（LIST01）の定義

LIST01 を管理するバックエンドサーバの名称（b001）を指定します。
RD エリアのページ長及びセグメントサイズを指定します。HiRDB ファイルシステム領域に list01 という HiRDB ファイルを作成します。HiRDB ファイル内のセグメント数を 1000 とします。

### （2） データベース構成変更ユティリティ（pdmod）の実行

**コマンドの入力例**

```
pdmod -a C:¥hirdb¥pdmod04
```

〔説明〕

-a：（1）で作成した制御文ファイルのファイル名を指定します。

# 5 プラグインの環境設定

プラグインの環境設定は，HiRDB の環境設定の後で実施します。この章では，プラグインの環境設定方法，バージョンアップ方法及び削除（アンインストール）方法について説明します。


5.1 **プラグインの環境設定の概要**

5.2 **プラグインのバージョンアップ**

5.3 **プラグインの削除**

# 5.1 プラグインの環境設定の概要

HiRDB のプラグインの環境を設定する方法について説明します。

## 5.1.1 環境設定手順

**実行者** HiRDB 管理者

ここでは，コマンドを使用したプラグインの環境設定方法について説明します。ここでの説明は，HiRDB の環境設定が終了している（HiRDB が既に稼働している）ことを前提として説明します。

プラグインの環境設定手順を次に示します。

**〈手順〉**

1. プラグインを組み込むのに必要なリソースの見積もり
2. 稼働中の HiRDB の終了
3. プラグインのインストール
4. HiRDB の開始
5. データディクショナリ LOB 用 RD エリア，ユーザ用 RD エリア及びユーザ LOB 用 RD エリアの追加※ 1
6. プラグインの登録
7. レジストリ機能の初期設定※ 2
8. HiRDB の終了
9. pdplugin オペランドの追加
10. HiRDB の開始
11. レジストリ情報の登録

注※ 1

データディクショナリ LOB 用 RD エリアは，既にストアドファンクション，ストアドプロシジャ又はプラグインを使用している場合は不要です。ユーザ用 RD エリア（ユーザ LOB 用 RD エリア）は，新規追加したプラグイン用に表を作成した場合に必要です。

注※ 2

使用するプラグインによっては不要な場合があります。

### (1) リソースの見積もり

プラグインを HiRDB に組み込む前に次に示すリソースを見積もる必要があります。

- プラグインを実行するために必要なメモリ所要量
- プラグインをインストールするときのディスク容量

プラグインを組み込むときに必要なリソースの見積もり方法については，該当するプラグインのマニュアルを参照してください。

### (2) HiRDB の終了

プラグインをセットアップする前に，pdstop コマンドで現在稼働している HiRDB を終了してください。

### (3) プラグインのインストール

プラグインをインストールします。インストール方法については，該当するプラグインのマニュアルを参照してください。

### (4) HiRDB の開始

pdstart コマンドで HiRDB を開始します。

### (5) RD エリアの追加

HiRDB にプラグインを登録する前に，データベース構成変更ユティリティ（pdmod）の create rdarea 文で必要な RD エリアを追加してください。追加する必要がある RD エリアを次に示します。

- ユーザ用 RD エリア※1
- ユーザ LOB 用 RD エリア※1
- データディクショナリ LOB 用 RD エリア※2（既にストアドプロシジャ，ストアドファンクション，プラグインを使用している場合は不要です）

RD エリアの追加方法については，「4.7　ユーザ用 RD エリアの作成」，「4.8　ユーザ LOB 用 RD エリアの作成」又は「4.9　データディクショナリ LOB 用 RD エリアの作成」を参照してください。

なお，既にデータベース環境を構築している場合には，プラグインインストール後の RD エリアの追加は不要です。

注※1
プラグイン用に別の表を作成し，その表を格納するための RD エリアを新しく用意したい場合に必要です。

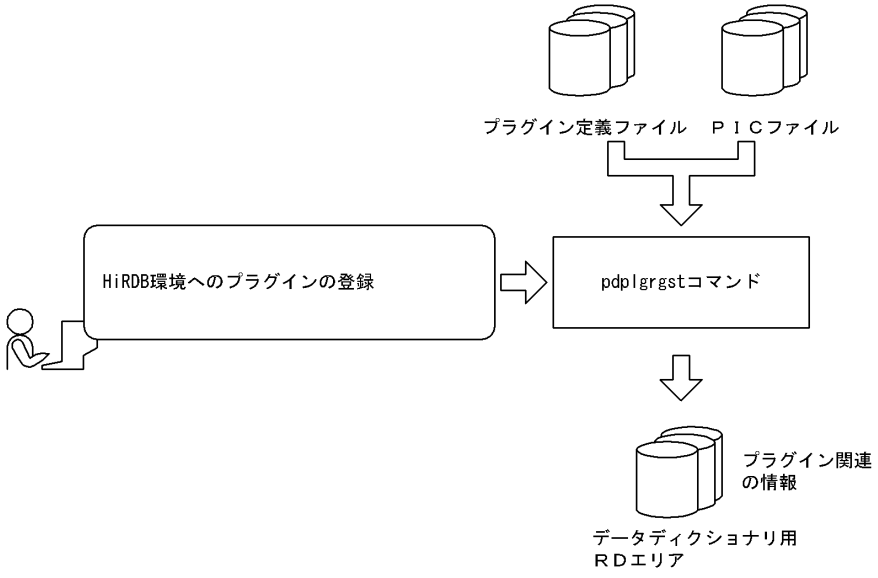
注※2
レジストリ機能初期設定ユティリティ（pdreginit）を実行する場合に，あらかじめ HiRDB がストアドプロシジャ機能を使用できるようにしておく必要があるため，この RD エリアが必要です。

### (6) プラグインの登録

pdplgrgst コマンドでプラグインを HiRDB に登録してください。pdplgrgst コマンドは，任意のサーバマシンから入力してください。

プラグインの登録の流れを次の図に示します。

**図 5-1 プラグインの登録の流れ**




### (a) pdplgrgst コマンドの入力形式

pdplgrgst コマンドの入力形式を次に示します。

pdplgrgst　**プラグイン定義ファイル名**　PIC **ファイル名**

HiRDB Text Search Plug-in **の場合の指定例**

- **データ型プラグインの場合：**
  pdplgrgst _phsgml.adt _phsgml.pic
  （カレントディレクトリが C:¥TSPlugin¥_phsgml¥etc の場合）
- **インデクス型プラグインの場合：**
  pdplgrgst _phngram.idx _phngram.pic
  （カレントディレクトリが C:¥TSPlugin¥_phngram¥etc の場合）

**注意事項**

- インデクス型プラグインを登録する場合，あらかじめ対応するデータ型プラグインが登録されていなければなりません。
- データ型プラグインとインデクス型プラグインは，必ず同じスキーマ内に登録してください。

### (b) プラグインの所有者

プラグインの所有者（プラグインが提供する抽象データ型，インデクス型及び関数の所有者）は MASTER になります。したがって，プラグインが提供する関数を呼び出す処理を SQL 文に記述する場合，認可識別子を省略できます。

MASTER **以外にする場合**

所有者を MASTER ではなく，pdplgrgst コマンドの実行者にできます。pdplgrgst コマンドに -u オプションを指定すると，プラグインの所有者は pdplgrgst コマンドの実行者（クライアント環境定義の PDUSER オペランドに指定した認可識別子）になります。ただし，この場合，次に示す注意があります。

**注意事項**

1. pdplgrgst コマンド実行者のスキーマが既に定義されている必要があります。
2. プラグインが抽象データ型及びインデクス型の両方を提供している場合は，必ず同じ所有者にしてください。
3. プラグインの削除又はバージョンアップは，プラグインの所有者しかできません。また，pdplgrgst コマンドに -u オプションを指定してプラグインを削除又はバージョンアップしてください。
4. プラグイン所有者のスキーマを削除すると，プラグインも同時に削除されます。この場合，次に示す作業が必要です。
   ・システム共通定義から pdplugin オペランドを削除します。
5. 複数のプラグインで，関数名及びパラメタ数が同じである関数を提供している場合，一方のプラグインを登録し，そのプラグインが提供する関数を呼び出す関数をユーザが定義した後，もう一方のプラグインを登録します。このとき，ユーザが定義した関数のパラメタ，又は戻り値のデータ型に抽象データ型を使用し，かつ，その関数をビュー定義に使用している場合は，プラグインの登録時にエラーとなります。この場合，その関数を使用したビュー表を削除した後，再度プラグインの登録を行う必要があります。

## (7) レジストリ機能の初期設定

プラグインによっては，レジストリ機能が必要な場合があります。その場合，レジストリ機能初期設定ユティリティ（pdreginit）の create rdarea 文で次に示す RD エリアを作成してください。ただし，既にプラグインでレジストリ機能を使用している場合は不要です。

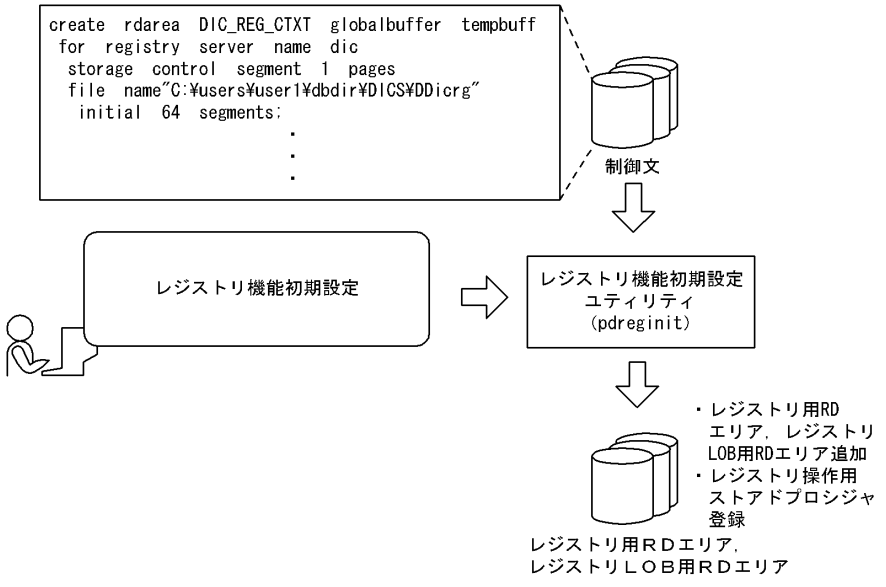
- レジストリ用 RD エリア
- レジストリ LOB 用 RD エリア

なお，レジストリ機能初期設定ユティリティ（pdreginit）は，すべてのプラグインを登録するまでの間で一度だけ実行します。

また，レジストリ用 RD エリア及びレジストリ LOB 用 RD エリアには，レジストリ情報が格納されます。どちらの RD エリアに格納されるかは，登録されるデータの長さによって自動的に決定されます。

レジストリ用 RD エリア，レジストリ LOB 用 RD エリアの作成の手順を次の図に示します。

**図 5-2 レジストリ用 RD エリア，レジストリ LOB 用 RD エリアの作成の手順**




### (8) HiRDB の終了

プラグインを使用できる状態にするため，いったん pdstop コマンドで HiRDB を正常終了させます。再度開始するまでは，登録したプラグインを使用した表の定義やインデクスの定義などは実行できません。

HiRDB 終了後，更新した RD エリアのバックアップを必ず取得してください。

### (9) pdplugin オペランドの追加

HiRDB が正常終了したら，システム共通定義に pdplugin オペランドを指定してください。pdplugin オペランドには，使用するプラグインの名称を指定します。

HiRDB/ パラレルサーバの場合，すべてのサーバマシン上のシステム共通定義に pdplugin オペランドを追加してください。追加漏れがあると HiRDB を開始できません。

### (10) HiRDB の開始

pdstart コマンドで HiRDB を開始します。

### (11) レジストリ情報の登録

レジストリ機能の初期設定の完了後，プラグインの必要に応じてレジストリ情報を登録します。登録すると，プラグイン及びレジストリ機能を使用できる状態になります。レジストリ情報の登録については，該当するプラグインのマニュアルを参照してください。

## 5.1.2 プラグイン使用時の注意

### (1) HiRDB の設定／開始時の状態とプラグインの使用可否

HiRDB（ユニット）の設定／開始時の状態とプラグインの使用可否を次の表に示します。

表 5-1 HiRDB（ユニット）の設定／開始時の状態とプラグインの使用可否

| プラグインの利用宣言（pdplugin オペランド） | プラグインの登録（pdplgrgst） | プラグインの初期化エラー | ユニット開始可否 | プラグイン使用可否 |
|---|---|---|---|---|
| なし | 登録済み | なし | ○ | × |
| | | あり | ○ | × |
| | 未登録 | なし | ○ | × |
| | | あり | ○ | × |
| あり | 登録済み | なし | ○ | ○ |
| | | あり | ○ | × |
| | 未登録 | なし | ○ | × |
| | | あり | ○ | × |

（凡例）

○：ユニットを開始できます。及びプラグインを利用できます。

×：ユニットを開始できません。及びプラグインを利用できません。

### (2) プラグイン初期化エラーが発生したとき

プラグインの初期化は，HiRDB 開始時に自動的に実行されます。システム共通定義に複数の pdplugin オペランドが記述され，一部のプラグインの初期化でエラーが発生した場合，ユニット内のすべてのプラグインが利用できません。

### (3) ユニット間でプラグインの使用可否が異なる場合

次に示す場合，ユニット間で利用できるプラグインが異なります。

- **ユニット間でシステム共通定義に記述されたプラグイン利用宣言（pdplugin オペランド）が異なる場合**
- **プラグイン初期化処理時にエラーが発生し，そのユニットでプラグインが利用できなくなった場合**

プラグインの呼び出しを伴う SQL 文で利用できるプラグインだけ呼び出した場合は成功しますが，一つでも利用できないプラグインが呼び出された場合は，SQL 文の実行は失敗します。

# 5.2 プラグインのバージョンアップ

HiRDB に導入済みのプラグイン（データ型プラグイン，インデクス型プラグイン）をバージョンアップする手順について説明します。プラグインのバージョンアップとは，次に示す内容を削除しないでプラグインのバージョンを上げることです。

- プラグインの提供するデータ型を使用した表やビュー表
- プラグインの提供するインデクス型を使用したインデクス
- プラグインの提供する関数

## 5.2.1 バージョンアップ時の注意

プラグインをバージョンアップする場合，次に示す注意が必要です。

- データ型プラグインとインデクス型プラグインのバージョンを合わせてください。
- データ型プラグインとインデクス型プラグインの両方をバージョンアップする場合，データ型プラグインを先にバージョンアップしてください。
- 新しいバージョンのプラグインを登録した場合，古いバージョンのプラグインに戻せません。
- データディクショナリ表の SQL_PLUGIN_ROUTINES 表及び SQL_PLUGIN_ROUTINE_PARAMS 表の SPECIFIC_NAME 列は更新されます。
- プラグインのバージョンアップ時にプラグインの所有者を変更すると，関数が無効になることがあります。この場合，ALTER ROUTINE で関数の SQL オブジェクトを再作成してください。

## 5.2.2 バージョンアップの手順

バージョンアップの手順を次に示します。

### （1）バックアップの取得

障害時に備えて，**データベース複写ユティリティ（pdcopy）**で次に示す RD エリアのバックアップを取得します。なお，データベース複写ユティリティ（pdcopy）に -M x オプションを指定してください。

- マスタディレクトリ用 RD エリア
- データディクショナリ用 RD エリア
- データディレクトリ用 RD エリア
- データディクショナリ LOB 用 RD エリア

バックアップの取得方法については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

### （2）HiRDB の終了

pdstop コマンドで HiRDB を正常終了させます。

### （3）必要なファイルの退避

次に示す場所にあるファイルのうち，必要なものを退避します。

- %PDDIR%¥plugin¥ **プラグイン名 のディレクトリ**

どのファイルを退避するかについては，該当するプラグインのマニュアルを参照してください。HiRDB がセットアップされたすべてのサーバマシンでファイルを退避してください。

### (4) 新バージョンのプラグインのインストール

HiRDB の各サーバマシンで，新しいバージョンのプラグインをインストールします。インストール方法については，各プラグイン提供のマニュアルを参照してください。

### (5) システム共通定義の変更

システム共通定義から，バージョンアップ対象のプラグインの pdplugin オペランドを削除します。システム共通定義ファイルがあるすべてのサーバマシンから削除してください。

### (6) 退避したファイルの復旧

HiRDB がセットアップされたすべてのサーバマシンで，(3) で退避したファイルを復旧してください。

### (7) HiRDB の開始

pdstart コマンドで HiRDB を開始します。

### (8) 新バージョンのプラグインの登録

pdplgrgst コマンドに -a オプションを指定して実行し，プラグインを更新登録します。

### (9) HiRDB の終了

pdstop コマンドで HiRDB を正常終了させます。

### (10) システム共通定義の変更

HiRDB が正常終了したら，システム共通定義に pdplugin オペランドを指定してください。pdplugin オペランドには，バージョンアップしたプラグインの名称を指定します。

HiRDB/ パラレルサーバの場合，すべてのサーバマシン上のシステム共通定義に pdplugin オペランドを追加してください。追加漏れがあると HiRDB を開始できません。

### (11) HiRDB の開始

pdstart コマンドで HiRDB を開始します。開始すると，バージョンアップ前に定義していた表，インデクスを使用できるようになります。さらに，バージョンアップ後のプラグインが新しい機能を提供する場合，その機能を利用できます。

# 5.3 プラグインの削除

ここでは，HiRDB に登録されたプラグインを削除する方法と手順について説明します。なお，プラグインの削除とは，次に示す内容を削除することです。

- ディクショナリに登録されたプラグインの定義情報
- プラグインが提供する関数，抽象データ型及びインデクス型

ただし，プラグインファイルセットを含むプラグイン提供のファイルは削除しません。プラグインを削除する手順を次に示します。

## (1) プラグインが提供する機能を使用したデータベース資源の削除

プラグインを削除する前に，次に示すデータベース資源を削除する必要があります。これらを削除するために発行する SQL 文を次の表に示します。

- 削除するプラグインの提供する抽象データ型を利用した表，ビュー表，関数，手続き，抽象データ型（ユーザが定義した抽象データ型の中で，プラグインが提供する抽象データ型を一つの属性として指定した場合）
- 削除するプラグインの提供するインデクス型を利用したインデクス
- 削除するプラグインの提供する関数を利用した関数，手続き

**表 5-2 データベース資源を削除するための SQL**

| 削除する対象となるデータベース資源 | 削除するための SQL |
|---|---|
| 表 | DROP TABLE |
| インデクス | DROP INDEX |
| ビュー表 | DROP VIEW |
| 関数 | DROP FUNCTION |
| 手続き | DROP PROCEDURE |
| 抽象データ型 | DROP DATA TYPE ※ |

注※ プラグインが提供するデータ型を削除してはいけません。

## (2) 登録したプラグインの削除

削除する個々のプラグインについて次に示すコマンドを実行します。

pdplgrgst -d **プラグイン定義ファイル名** PIC **ファイル名**

**注意事項**

1. データ型プラグインを削除する場合で，そのデータ型のインデクス機能を提供するインデクス型プラグインも登録されているときは，インデクス型プラグインを先に削除する必要があります。
2. プラグインの所有者が MASTER 以外の場合，プラグインを削除するときに次に示すことに注意してください。

- プラグインの削除はプラグインの所有者だけができます。クライアント環境定義の PDUSER オペランドにプラグインの所有者の認可識別子とパスワードを指定してください。そして，pdplgrgst コマンドを実行するときに -u オプションを指定してください。
- プラグイン所有者のスキーマを削除した場合，プラグインも同時に削除されます。プラグインが削除された後の処理については，(3) 以降を参照してください。

### (3) レジストリの削除

レジストリに登録した情報を削除する方法については，各プラグインで提供しているマニュアルを参照してください。

### (4) HiRDB の終了

pdstop コマンドで HiRDB を正常終了させます。

### (5) システム共通定義の変更

HiRDB が正常終了したら，システム共通定義から pdplugin オペランドを削除します。

### (6) プラグインのアンインストール

プラグインをサーバマシンからアンインストールします。アンインストール方法については，該当するプラグインでの手順に従ってください。

# 6 データベースの作成

この章では，スキーマ，表，インデクスを作成し，データを格納するまでの方法について説明します。


6.1 **データベース作成の概要**

6.2 **横分割表の作成**

6.3 LOB **列を定義した表の作成**

6.4 **プラグインが提供する抽象データ型を定義した表の作成**

6.5 **ユーザが定義した抽象データ型を定義した表の作成**

6.6 **インデクス一括作成中に発生したエラーの対処方法**

6.7 **同期点指定のデータロード実行中にユティリティが異常終了したときの対処方法**

# 6.1 データベース作成の概要

ここでは，データベース（表及びインデクス）を作成する前に理解しておくことについて説明します。ここで説明する項目を次に示します。

- データベースの作成前に必要な作業
- データベースの作成手順
- データベースの更新ログ取得方式
- ユニークインデクスを定義した表にデータロードする場合の注意
- 大量のデータをロードする場合（同期点指定のデータロード）
- 横分割表にデータをロードする場合（分割入力データファイルの作成）
- 自動採番機能を使用したデータロード
- 入力データファイル UOC
- 不要な RD エリアの確認

## 6.1.1 データベースを作成する前に必要な作業

**実行者　HiRDB 管理者**

データベースを作成する前に必要な作業について説明します。

### (1) クライアント環境定義の設定

次に示すクライアント環境定義を設定してください。クライアント環境定義の設定方法については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

- PDHOST
- PDUSER
- PDNAMEPORT

### (2) パスワードの変更

HiRDB 管理者用の認可識別子のパスワードが，認可識別子と同じ文字列になっている場合，定義系 SQL の GRANT でパスワードを変更してください。データベース定義ユティリティ（pddef）又は HiRDB SQL Executer で次に示す SQL を実行します。

```
GRANT DBA TO HiRDB管理者用の認可識別子 IDENTIFIED BY 新しいパスワード;
```

### (3) スキーマの定義

定義系 SQL の CREATE SCHEMA でスキーマを定義します。CREATE SCHEMA は，データベース定義ユティリティ（pddef）又は HiRDB SQL Executer で実行してください。なお，1 ユーザに対して 1 個のスキーマを定義します。

### (4) HiRDB 管理者以外がデータベースを作成する場合

HiRDB 管理者以外がデータベースを作成する場合，ユーザ権限を与える作業が必要になります。定義系 SQL の GRANT で，データベースを作成するユーザに必要なユーザ権限を与えてください。必要なユーザ権限を次に示します。

- CONNECT 権限
- スキーマ定義権限

- RD エリア利用権限

ユーザ権限については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

（例）

表を作成するユーザ（認可識別子：USER002，パスワード：HIRDB002）に，CONNECT 権限，スキーマ定義権限，RD エリア利用権限（RD エリア名：RDAREA01）を与えます。

```
GRANT CONNECT TO USER002 IDENTIFIED BY HIRDB002;
GRANT SCHEMA TO USER002;
GRANT RDAREA RDAREA01 TO USER002;
```

### (5) データの変換方式の設定

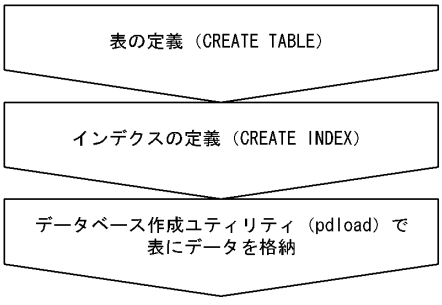
データをデータベースに格納するとき，データの形式を変換する機能があります。次に示す機能を適用するかどうかを検討してください。これらの機能については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

- 空白変換機能
- DECIMAL 型の符号正規化機能

## 6.1.2 データベースの作成手順

データベースの作成手順を次の図に示します。

**図 6-1 データベースの作成手順**




注

CREATE TABLE 及び CREATE INDEX は次に示すどちらかの方法で実行します。

- データベース定義ユティリティ（pddef）
- HiRDB SQL Executer

## 6.1.3 データベースの更新ログ取得方式

**データベース作成ユティリティ（pdload）**で表にデータを格納するときに，データベースの更新ログ取得方式を指定します。データベースの更新ログ取得方式は，データベース作成ユティリティ（pdload）の -l オプションで指定します。

### (1) データベースの更新ログ取得方式の種類

データベースの更新ログ取得方式には，次に示す 3 種類のモードがあります。

- **ログ取得モード**
  ロールバック及びロールフォワードに必要なデータベース更新ログを取得します。データ件数が少ない

場合に使用します。

- **更新前ログ取得モード**
  ロールバックに必要なデータベース更新ログだけを取得します。データ件数が多いときに使用します。
- **ログレスモード**
  データベース更新ログを取得しません。そのため，データロードの処理時間が最も掛かりません。一つの RD エリアに一つの表だけ（分割格納している場合は一つの横分割表）を格納している場合で，関連するインデクスも一つの RD エリア内にあるときに使用します。

これらのモードの機能詳細については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

### (2) データの格納先がユーザ LOB 用 RD エリアの場合

データの格納先がユーザ LOB 用 RD エリアの場合，CREATE TABLE の RECOVERY オペランドでデータベースの更新ログ取得方式を指定します。

ユーザ LOB 用 RD エリアのデータベースの更新ログ取得方式（CREATE　TABLE の RECOVERY オペランド指定値）は，pdload の -l オプションの指定値によって変わることがあります。pdload の -l オプションの指定値によって変わるユーザ LOB 用 RD エリアのデータベースの更新ログ取得方式を次の表に示します。

**表 6-1　pdload の -l オプションの指定値によって変わるユーザ LOB 用 RD エリアのデータベースの更新ログ取得方式**

| pdload の -l オプションの指定値 | CREATE TABLE の RECOVERY オペランドの指定値 | | |
|---|---|---|---|
| | ALL | PARTIAL | NO |
| a（ALL 指定相当） | ALL | PARTIAL | NO |
| p（PARTIAL 指定相当） | PARTIAL ※ | PARTIAL ※ | NO |
| n（NO 指定相当） | NO | NO | NO |

（凡例）

ALL：ログ取得モード
PARTIAL：更新前ログ取得モード
NO：ログレスモード
例えば，CREATE TABLE の RECOVERY オペランドに PARTIAL を指定し，pdload の -l オプションに n を指定した場合，NO（ログレスモード）がユーザ LOB 用 RD エリアに設定されます。

注※

プラグインが出力するログの場合は，ALL（ログ取得モード）が仮定されます。

### (3) モードの選択基準

基本的には省略値である更新前ログ取得モードを選択してください。ただし，次に示す条件を満たすような場合はほかのモードの選択を検討してください。

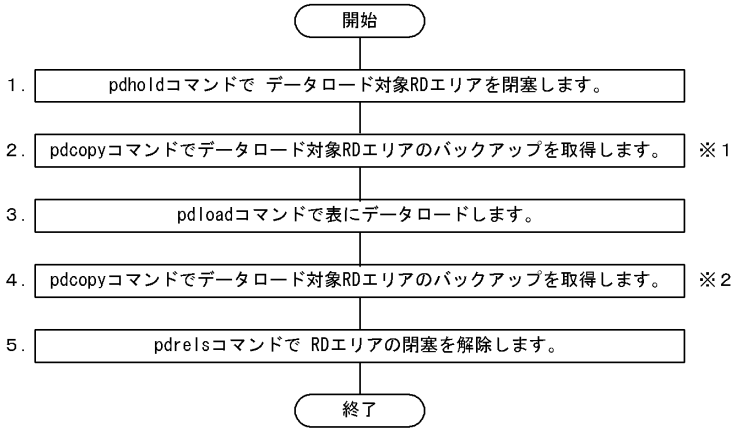
| 条件 | 選択するモード |
|---|---|
| RD エリアにデータロード対象表（インデクス）だけを格納していて，かつ初期ロードである | ログレスモード |
| 入力データ量が多く，データロードに時間が掛かる | |

| 条件 | 選択するモード |
|---|---|
| 入力データ量が少ない | ログ取得モード |

## (4) 運用方法の違い

選択したモードによってデータロードするときの運用が異なります。運用方法の違いを次の図に示します。

**図 6-2 データベースの更新ログ取得方式による運用方法の違い（データロード）**




注※1

ログレスモードを選択したときに必要な操作です。ログレスモードの pdload コマンドが異常終了した場合，このバックアップを使用して RD エリアを回復します。ただし，「(5) データロードの前にバックアップを取得しないでよい場合」で説明している条件を満たすときはバックアップを取得する必要はありません。

ただし，更新ログ取得方式に関係なく，プラグインインデクスを定義した表にインデクス一括作成モードで追加データロードをする場合はバックアップを取得してください。理由は，pdload コマンドが異常終了した場合のデータベース回復には既存データ部分を含め，全プラグインインデクスの再作成が必要となり，データベース回復に長時間必要となるためです。

注※2

更新前ログ取得モード又はログレスモードを選択したときに必要な操作です。ここでバックアップを取得しないと，pdrstr コマンドで RD エリアを回復する必要が生じた場合，RD エリアを最新の状態に回復できません（データロード実行後の反映処理を回復できません）。RD エリアはデータロード実行前の状態にしか回復できません。

**補足事項**

更新前ログ取得モード又はログレスモードを選択した場合，前記の手順 1 ～ 4 の間はデータロード対象 RD エリアを閉塞したままにしてください。手順 4 でバックアップを取得する前に RD エリアの内容が更新された場合，pdrstr コマンドで RD エリアを回復する必要が生じたときにその更新内容を回復できません。RD エリアはデータロード実行前の状態にしか回復できません。pdrstr コマンドで RD エリアを回復するとき，入力情報のシステムログ中に更新前ログ取得モード又はログレスモードで取得したログが入っていると pdrstr コマンドがエラーになります。

## (5) データロードの前にバックアップを取得しないでよい場合

ログレスモードでデータロードを実行する場合は，データロードの実行前にバックアップを取得する必要

があります。ただし，次表に示す 1，2 の条件のうちどちらかを満たす場合は，pdload コマンドが異常終了したときに RD エリアの状態をデータロード実行前の状態に戻すことができるため，バックアップの取得を省略できます。

| 項番 | 条件 | | 障害発生時の RD エリア回復方法 |
|---|---|---|---|
| 1 | 初期ロードの場合 | データロード対象 RD エリアにはデータロード対象の表，及びその表のインデクスだけを格納している場合 | データロード対象 RD エリアをデータベース構成変更ユティリティ（pdmod コマンド）で再初期化した後に，再度データロードすると回復できます。 |
| | 作成モードでデータロードを実行する場合 | | |
| 2 | データロード対象 RD エリアにデータロード対象表（インデクス）以外の表（インデクス）がある場合 | バックアップとシステムログを使用してデータロード実行前の状態に RD エリアを回復できる場合 | pdclose コマンドでデータロード対象 RD エリアをクローズした後に，pdlogswap コマンドでシステムログファイルをスワップして，データベース回復ユティリティ（pdrstr コマンド）にここまでのシステムログを入力すれば回復できます。 |
| | 追加モードでデータロードを実行する場合 | | |

注

項番 2 の条件の場合は，バックアップを取得した方が RD エリアを回復するときの運用が簡単なため，基本的にはバックアップを取得することをお勧めします。特に，インデクス一括作成モードの pdload が異常終了した場合は，ログ取得モードであっても更新前ログ取得モードであっても，ロールバックではインデクスは回復されません。pdload が異常終了して，すぐにデータロード前の状態に回復する必要がある場合は必ずバックアップを取得してください。

## 6.1.4 ユニーク属性のインデクスを定義した表にデータロードする場合の注意

**主キーインデクス（PRIMARY インデクス）**，又は**ユニークインデクス（UNIQUE 指定のインデクス）**を定義した表にデータロードをする場合は，次に示す注意が必要です。

- **入力データファイル中にキー値が重複するデータがある場合，インデクス一括作成モードでデータロードをしないでください。**

インデクス一括作成モードでデータロードをすると，データを表に格納し，インデクスのキー情報をインデクス情報ファイルに出力します。この段階ではキー値の重複チェックはされません。キー値の重複チェックは，その後のインデクスデータの格納段階でチェックされます。キー値が重複していると，インデクスの作成処理はロールバックされますが，データは既に格納されています（コミットされて元に戻りません）。この場合，バックアップを使用して RD エリアを回復する必要があります。

したがって，**キー値の重複データがある入力データファイルを使用してデータロードをする場合は，必ずインデクス更新モードを指定してください。**このモードはデータを格納するごとにインデクスを更新するのでキー値の重複をすぐに検知し，該当するデータはデータベースに格納しません。

インデクス一括作成モード及びインデクス更新モードは，データベース作成ユティリティ（pdload）の -i オプションで指定します。なお，**省略値がインデクス一括作成モードになっているので注意してください。**

## 6.1.5 大量のデータをロードする場合（同期点指定のデータロード）

大量のデータを表にロードする場合，**同期点指定のデータロード**を実施するかどうかを検討してください。

通常，データロード処理では全データの格納処理を完了するまでトランザクションを決着できません。このため，データベース作成ユティリティ実行中はシンクポイントダンプを有効化できません。したがって，大量データのロード中に HiRDB が異常終了すると，HiRDB の再開始処理に長い時間を必要とします。こ

れを防ぐために，データロード時に任意の件数で同期点を設定してトランザクションを決着できます。これを同期点指定のデータロードといいます。

同期点指定のデータロードをするには，データベース作成ユティリティの option 文で**同期点行数**（何件データを格納したら同期点を取得するか）を指定してください。

**注意事項**

1. この機能を適用しない場合よりも同期点処理が実行される分，処理性能が低下します。
2. ユティリティが異常終了したとき，そのタイミングによって対処方法が異なります。異常終了時の対処方法については，「6.7　同期点指定のデータロード実行中にユティリティが異常終了したときの対処方法」を参照してください。特に，インデクス一括作成モードでデータロードをしたときにユティリティが異常終了すると，対処方法が複雑になるため注意してください。
3. 同期点のたびに新しいページからデータの格納を開始するので，この機能を適用しない場合よりも格納ページ数が多く必要になります。

### 6.1.6　横分割表にデータをロードする場合（分割入力データファイルの作成）

横分割表にデータをロードする場合，格納する RD エリア単位に入力データファイルを分割してデータロードを並列実行すると，データロードに掛かる時間を短縮でき，表の占有時間が短縮できます。制御情報ファイルに src_work 文を指定してデータベース作成ユティリティ（pdload）を実行すると，ユーザが作成した入力データファイルから，RD エリア単位の入力データファイルが作成できます。これによって，RD エリア単位にデータロードを実行できます。このとき，データベース作成ユティリティ（pdload）が作成するファイルを**分割入力データファイル**といいます。分割入力データファイルを作成する場合のオプション及び制御文については，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

### 6.1.7　自動採番機能を使用したデータロード

順序数生成子を使用すると，自動的に採番ができます。これを自動採番機能といいます。データロード時に，順序数生成子が生成した順序番号を，表の列に格納できます。ここでは，順序番号の取得方式と格納方式の選択基準について説明します。

なお，自動採番機能についてはマニュアル「HiRDB Version 8 UAP 開発ガイド」を，自動採番機能を使用したデータロードの詳細についてはマニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

#### (1) 順序番号の取得方式の選択基準

順序番号の取得方式には，次の 3 種類があります。

**全数一括取得方式**：
　データロード完了後に一括して順序数生成子の値を使用した順序番号にします。

**指定単位取得方式**：
　指定した単位ごとに順序番号を取得しながらデータロードします。

**バッファ単位取得方式**：
　入力バッファに読み込める行数分の順序番号を取得しながらデータロードします。

順序番号の取得方式は，次の表に示す特徴を考慮して選択してください。

表 6-2 順序番号の取得方式ごとの特徴

| 検討項目 | 特徴 | | |
|---|---|---|---|
| | 全数一括取得方式 | 指定単位取得方式 | バッファ単位取得方式 |
| 正常時の欠番発生 | 発生しません。 | データロードした行数が指定した単位の倍数でなければ，欠番が発生します。 | 発生しません。※2 |
| ロールバック時の大量の欠番発生 | 発生しません。 | 現在値はロールバックが発生しても回復されないため，大量の欠番が発生します。 | 現在値はロールバックが発生しても回復されないため，大量の欠番が発生します。 |
| 順序数生成子への採番要求時の通信オーバヘッド※1 | データロードのコミットの回数分しか採番要求をしないため，採番処理による性能への影響は小さくなります。 | 取得する単位を大きくすることで，採番要求の回数を抑え，性能への影響を小さくできます。 | 入力バッファ長を大きくすることで，採番要求の回数を抑え，性能への影響を小さくできます。ただし，1回に取得する単位は入力バッファ長と列の定義長から算出した行長で決まるため，行長が大きいと入力バッファに多くのメモリを必要とします。 |
| 同じ順序数生成子を使用する UAP との同時実行 | 同時実行はできません。データロード中は順序数生成子に排他が掛かります。 | 同時実行できます。 | 同時実行できます。 |
| RD エリア単位のデータロードの並列実行 | 同時実行はできません。データロード中は順序数生成子に排他が掛かります。 | 並列実行できます。 | 並列実行できます。 |

注※1

HiRDB/ パラレルサーバの場合，データベース作成ユティリティが入力データを読み込むサーバと順序数生成子が定義されたサーバが異なるとき，順序番号の取得時に通信が発生します。そのため，採番要求が頻繁に行われると，通信回数が多くなり，データロードの性能に影響します。

注※2

順序番号の格納方式が列データ全置換以外の場合は，欠番が発生する可能性があります。

## (2) 順序番号の格納方式の選択基準

順序番号の格納方式には，次の 3 種類があります。それぞれの選択基準について説明します。

**列データ全置換**：

順序番号を格納する列に対して，入力データファイル中の該当する列データをすべて順序番号に置き換えます。該当する列の値に，すべて新しく番号を振り直す場合に選択します。

**列データー部置換**：

順序番号を格納する列に対して，入力データファイル中の該当する列データのうち，指定した置換条件に一致するデータだけ順序番号に置き換えます。例えば，入力データファイルが DAT 形式，又は拡張 DAT 形式で，NULL 値の部分だけ順序番号に置き換える場合などに選択します。

**列データ追加**：

順序番号を格納する列に対応するデータが入力データファイル中にない場合，順序番号を入力データとして追加します。番号を格納する列を新しく追加する場合に選択します。なお，入力データファイルがバイナリ形式の場合，この方式は指定できません。

### 6.1.8 入力データファイル UOC

任意に作成したプログラムで，データロードするデータを編集できます。編集されたデータは，直接 pdload に渡されます。そのため，入力ファイルを編集するプログラムで，いったんワークファイルを作成しない形でデータロードができます。

データを編集するためにユーザが作成したプログラムを UOC**（ユーザオウンコーディング）**といいます。UOC は，データベースに格納したいデータファイルが pdload の入力データファイルのフォーマットと異なる場合や，データベースに格納したいデータが HiRDB の文字コードと異なる場合など，入力データの編集が必要なときに使用できます。

### 6.1.9 不要な RD エリアの削除

データベースを作成後，ディクショナリ表の SQL_RDAREAS 表を検索して，表やインデクスが定義されなかったユーザ用 RD エリア又は LOB 列が定義されなかったユーザ LOB 用 RD エリアがないかどうか確認することをお勧めします。もし不要な RD エリアがあれば，削除できます。不要な RD エリアを削除することで，ディスク所要量を削減できます。

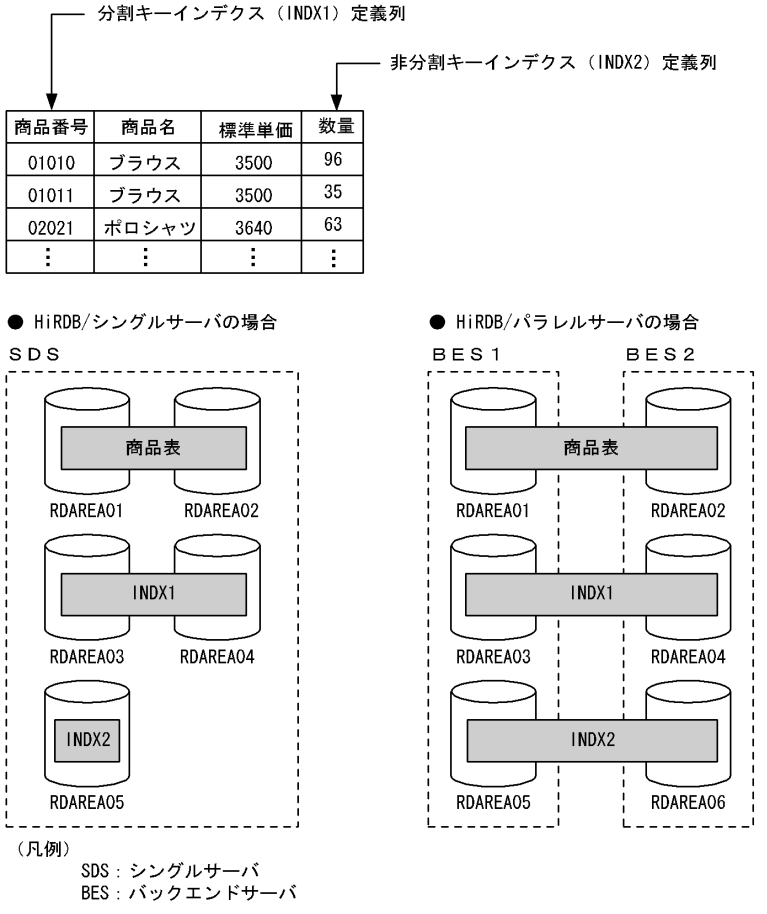
ディクショナリ表の検索方法と SQL_RDAREAS 表についてはマニュアル「HiRDB Version 8 UAP 開発ガイド」を，RD エリアの削除方法については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

# 6.2 横分割表の作成

商品表を作成します。商品表の作成条件を次に示します。

- 商品表を横分割します。ユーザ用 RD エリア RDAREA01 ～ RDAREA02 に商品表を格納します。
- 商品表に分割キーインデクス（INDX1）を定義します。ユーザ用 RD エリア RDAREA03 ～ RDAREA04 に INDX1 を格納します。
- 商品表に非分割キーインデクス（INDX2）を定義します。ユーザ用 RD エリア RDAREA05 に INDX2 を格納します。HiRDB/ パラレルサーバの場合は，ユーザ用 RD エリア RDAREA05 ～ RDAREA06 に INDX2 を格納します。
- RDAREA01 ～ RDAREA06 にはデータロード対象表（インデクス）だけを格納していて，初期ロードとします。
- データロードをするときにインデクスを一括作成（省略値）します。
- ログレスモードでデータロードをします。

分割キーインデクス及び非分割キーインデクスについては，「13.3　インデクスの横分割」を参照してください。



## (1) 商品表の定義

CREATE TABLE で商品表を定義します。定義例を次に示します。

### (a) キーレンジ分割の場合

**格納条件指定**

```
CREATE TABLE 商品表
(商品番号 CHAR(5),
 商品名 NCHAR(15),
 標準単価 INTEGER,
 数量 INTEGER
)IN ((RDAREA01)商品番号<='10000',(RDAREA02));
```

**境界値指定の場合**

```
CREATE TABLE 商品表
(商品番号 CHAR(5),
 商品名 NCHAR(15),
 標準単価 INTEGER,
 数量 INTEGER
 )PARTITIONED BY 商品番号
 IN ((RDAREA01)'10000',(RDAREA02));
```

### (b) フレキシブルハッシュ分割，FIX ハッシュ分割の場合

```
CREATE TABLE 商品表
(商品番号 CHAR(5),
 商品名 NCHAR(15),
 標準単価 INTEGER,
 数量 INTEGER
 )[FIX]※ HASH HASH6 BY 商品番号
 IN (RDAREA01,RDAREA02);
```

注※　FIX ハッシュ分割の場合に指定します。

## (2) インデクスの定義

CREATE INDEX で商品表にインデクスを定義します。定義例を次に示します。

### (a) HiRDB/ シングルサーバの場合

```
CREATE INDEX INDX1 ON 商品表 (商品番号)
    IN ((RDAREA03),(RDAREA04));
CREATE INDEX INDX2 ON 商品表 (数量)
    IN (RDAREA05);
```

### (b) HiRDB/ パラレルサーバの場合

```
CREATE INDEX INDX1 ON 商品表 (商品番号)
    IN ((RDAREA03),(RDAREA04));
CREATE INDEX INDX2 ON 商品表 (数量)
    IN ((RDAREA05),(RDAREA06));
```

## (3) 表へのデータの格納

データベース作成ユティリティ（pdload）で表にデータを格納します。格納手順を次に示します。

**〈手順〉**

1. pdhold コマンドで，データロード対象 RD エリア（RDAREA01 ～ RDAREA05）を閉塞します。HiRDB/ パラレルサーバの場合は RDAREA01 ～ RDAREA06 を閉塞します。
2. pdload コマンドで，入力データファイルを表にデータロードします。RD エリアにはデータロード対象表（インデクス）だけを格納していて，かつ初期ロードのため，データベースの更新ログ取得方式にログレスモードを選択します。また，インデクスの作成方法にインデクス一括作成モード

（省略値）を選択します。pdload コマンドに指定するオプションについては，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

3. ログレスモードで pdload コマンドを実行しているため，データロード対象 RD エリアのバックアップを取得します。RD エリア単位のバックアップの取得方法については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。
4. pdrels コマンドで，データロード対象 RD エリアの閉塞を解除します。

上記のコマンドとユティリティの詳細及びこれらのコマンドとユティリティの実行結果の確認方法については，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

**補足事項**

- ログレスモードで pdload コマンドを実行するため，前記の手順 1 ～ 3 の間はデータロード対象 RD エリアを閉塞したままにしてください。
- 改竄防止表に対して pdload コマンドでデータロードするとき，-d オプションは使用できません。
- インデクス一括作成中にエラーが発生した場合の対処方法については，「6.6　インデクス一括作成中に発生したエラーの対処方法」を参照してください。

## （4）データの格納状態の確認

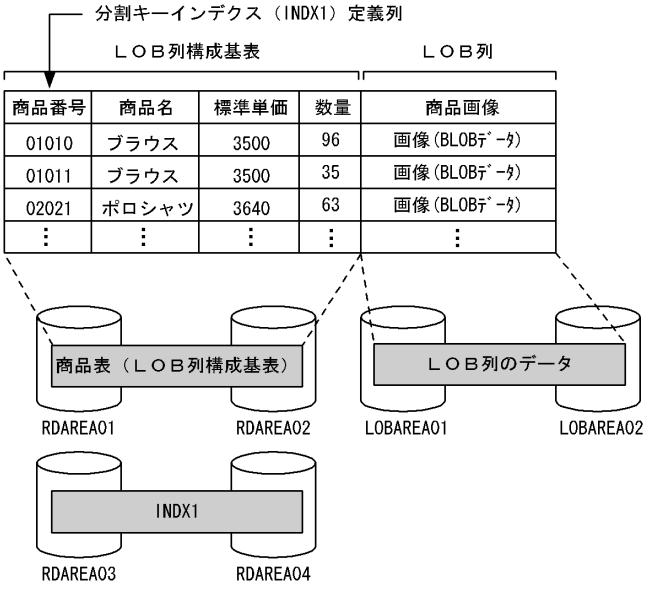
データロードをした場合は，運用を開始する前にデータベース状態解析ユティリティ（pddbst）を実行して，データの格納状態を確認することをお勧めします。設計どおりにデータベースを作成できたかどうかを確認できます。データベース状態解析ユティリティ（pddbst）を実行すると，次に示す情報を取得できます。

- ユーザ用 RD エリア単位のデータの格納状態
- 表又はインデクス単位のデータの格納状態

# 6.3 LOB 列を定義した表の作成

商品表を作成します。商品表の作成条件を次に示します。

- 商品表を横分割します。ユーザ用 RD エリア RDAREA01 ～ RDAREA02 に商品表の LOB 列構成基表を格納します。
- ユーザ LOB 用 RD エリア LOBAREA01 ～ LOBAREA02 に LOB 列のデータを格納します。
- 商品表に分割キーインデクス（INDX1）を定義します。ユーザ用 RD エリア RDAREA03 ～ RDAREA04 に INDX1 を格納します。
- RDAREA01 ～ RDAREA04，LOBAREA01 ～ LOBAREA02 にはデータロード対象表（インデクス）だけを格納していて，初期ロードとします。
- データロードをするときにインデクスを一括作成（省略値）します。
- ログレスモードでデータロードをします。




**補足事項**

- 一つのユーザ LOB 用 RD エリアには，表中の一つの LOB 列だけを格納します。また，一つの表に複数の LOB 列がある場合には，それぞれ別のユーザ LOB 用 RD エリアに格納する必要があります。
- LOB 列が横分割表の場合には，LOB 列ごとにユーザ LOB 用 RD エリアと，表を格納しているユーザ用 RD エリアの数を 1 対 1 に対応させる必要があります。

## (1) 商品表の定義

CREATE TABLE で商品表を定義します。定義例を次に示します。

### (a) キーレンジ分割の場合

**格納条件指定**

```
CREATE TABLE 商品表
(商品番号 CHAR(5),
```

```
 商品名 NCHAR(15),
 標準単価 INTEGER,
 数量 INTEGER,
 商品画像 BLOB(64K) IN ((LOBAREA01),(LOBAREA02))
)IN ((RDAREA01) 商品番号<='10000',(RDAREA02));
```

**境界値指定**

```
CREATE TABLE 商品表
(商品番号 CHAR(5),
 商品名 NCHAR(15),
 標準単価 INTEGER,
 数量 INTEGER,
 商品画像 BLOB(64K) IN ((LOBAREA01),(LOBAREA02))
)PARTITIONED BY 商品番号
IN ((RDAREA01)'10000',(RDAREA02));
```

**(b) フレキシブルハッシュ分割，FIX ハッシュ分割の場合**

```
CREATE TABLE 商品表
(商品番号 CHAR(5),
 商品名 NCHAR(15),
 標準単価 INTEGER,
 数量 INTEGER,
 商品画像 BLOB(6000) IN ((LOBAREA01),(LOBAREA02))
)[FIX]※ HASH HASH6 BY 商品番号
 IN (RDAREA01,RDAREA02);
```

注※　FIX ハッシュ分割の場合に指定します。

## (2) インデクスの定義

CREATE INDEX で商品表にインデクスを定義します。定義例を次に示します。

```
CREATE INDEX INDX1 ON 商品表 (商品番号)
    IN ((RDAREA03),(RDAREA04));
```

## (3) 表へのデータの格納

データベース作成ユティリティ（pdload）で表にデータを格納します。格納手順を次に示します。

**〈手順〉**

1. pdhold コマンドで，データロード対象 RD エリア（RDAREA01 ～ RDAREA04，LOBAREA01 ～ LOBAREA02）を閉塞します。
2. pdload コマンドで，入力データファイルを表にデータロードします。RD エリアにはデータロード対象表（インデクス）だけを格納していて，かつ初期ロードのため，データベースの更新ログ取得方式にログレスモードを選択します。また，インデクスの作成方法にインデクス一括作成モード（省略値）を選択します。pdload コマンドに指定するオプションについては，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。
3. ログレスモードで pdload コマンドを実行しているため，データロード対象 RD エリアのバックアップを取得します。RD エリア単位のバックアップの取得方法については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。
4. pdrels コマンドで，データロード対象 RD エリアの閉塞を解除します。

上記のコマンドとユティリティの詳細及びこれらのコマンドとユティリティの実行結果の確認方法については，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

**補足事項**

- ログレスモードで pdload コマンドを実行するため，前記の手順 1 ～ 3 の間はデータロード対象 RD

エリアを閉塞したままにしてください。
- 改竄防止表に対して pdload コマンドでデータロードするとき，-d オプションは使用できません。
- インデクス一括作成中にエラーが発生した場合の対処方法については，「6.6　インデクス一括作成中に発生したエラーの対処方法」を参照してください。

### (4) データの格納状態の確認

データロードをした場合は，運用を開始する前にデータベース状態解析ユティリティ（pddbst）を実行して，データの格納状態を確認することをお勧めします。設計どおりにデータベースを作成できたかどうかを確認できます。データベース状態解析ユティリティ（pddbst）を実行すると，次に示す情報を取得できます。

- ユーザ用 RD エリア又はユーザ LOB 用 RD エリア単位のデータの格納状態
- 表又はインデクス単位のデータの格納状態

### (5) 補足事項

LOB 列を定義した表にデータロードする場合，LOB 列構成基表と LOB データを別々にデータロードすることもできます。このときの手順を次に示します。

なお，データベースの更新ログ取得方式にログレスモードを，インデクスの作成方法にインデクス一括作成モード（省略値）を選択したとします。

**〈手順〉**

1. pdhold コマンドで，データロード対象 RD エリア（RDAREA01 ～ RDAREA04，LOBAREA01 ～ LOBAREA02）を閉塞します。
2. pdload コマンドで，入力データファイルを表（LOB 列構成基表及びインデクス）にデータロードします。このときのデータロード対象 RD エリアは RDAREA01 ～ RDAREA04 になります。このとき，LOB 中間ファイルに LOB 列のデータロード時に必要な情報を出力するようにしてください。pdload コマンドに指定するオプションについては，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。
3. pdload コマンドで，ユーザ LOB 用 RD エリア LOBAREA01 ～ LOBAREA02 にデータロードします。このとき，LOB 入力ファイルと 2. で作成した LOB 中間ファイルを指定します。
4. ログレスモードで pdload コマンドを実行しているため，データロード対象 RD エリアのバックアップを取得します。RD エリア単位のバックアップの取得方法については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。
5. pdrels コマンドで，データロード対象 RD エリアの閉塞を解除します。

上記のコマンドとユティリティの詳細及びこれらのコマンドとユティリティの実行結果の確認方法については，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

# 6.4 プラグインが提供する抽象データ型を定義した表の作成

ここでは，プラグインが提供する抽象データ型（SGMLTEXT 型，XML 型）を定義した表の作成について説明します。

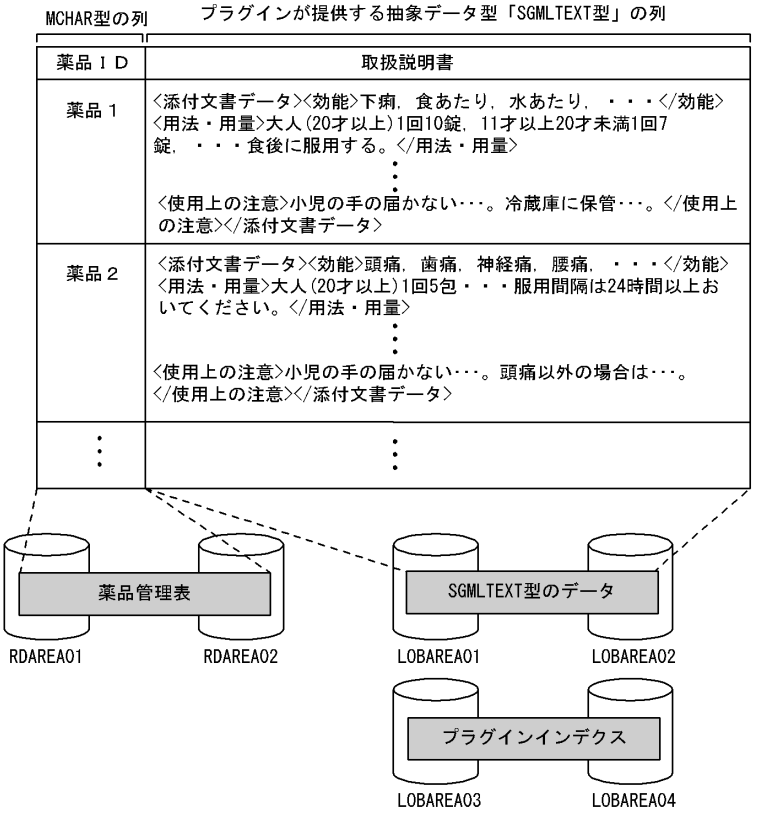
SGMLTEXT 型を使用する場合には HiRDB Text Search Plug-in，XML 型を使用する場合には HiRDB XML Extension が必要です。プラグインの環境設定については，「5. プラグインの環境設定」を参照してください。なお，プラグインの所有者は MASTER にしてください。

## 6.4.1 SGMLTEXT 型

ここでは，HiRDB Text Search Plug-in が提供する抽象データ型（SGMLTEXT 型）を定義した表の作成方法について説明します。

ここでは，薬品管理表を作成します。薬品管理表の作成条件を次に示します。

- 薬品管理表を横分割します。ユーザ用 RD エリア RDAREA01 ～ RDAREA02 に薬品管理表の LOB 列構成基表を格納します。
- ユーザ LOB 用 RD エリア LOBAREA01 ～ LOBAREA02 に SGMLTEXT 型の列のデータを格納します。
- ユーザ LOB 用 RD エリア LOBAREA03 ～ LOBAREA04 にプラグインインデクスを格納します。
- RDAREA01 ～ RDAREA02，LOBAREA01 ～ LOBAREA04 にはデータロード対象表（インデクス）だけを格納していて，初期ロードとします。
- データロードをするときにインデクスを一括作成（省略値）します。
- ログレスモードでデータロードをします。




〔説明〕

薬品 ID（MCHAR 型）をユーザ用 RD エリアに格納し，取扱説明書（SGMLTEXT 型）をユーザ LOB 用 RD エリアに格納します。

## (1) 薬品管理表の定義

CREATE TABLE で薬品管理表を定義します。定義例を次に示します。

### (a) キーレンジ分割の場合

**格納条件指定**

```
CREATE TABLE 薬品管理表(
薬品ID MCHAR(15),
取扱説明書 SGMLTEXT          …1
  ALLOCATE(SGMLTEXT IN((LOBAREA01),(LOBAREA02)))  …2
  PLUGIN'<DTD>medicine.dtd</DTD>'       …3
  )IN((RDAREA01)薬品ID<='薬品10',(RDAREA02));    …4
```

**境界値指定**

```
CREATE TABLE 薬品管理表(
薬品ID MCHAR(15),
取扱説明書 SGMLTEXT          …1
  ALLOCATE(SGMLTEXT IN((LOBAREA01),(LOBAREA02)))  …2
  PLUGIN'<DTD>medicine.dtd</DTD>'       …3
  )PARTITIONED BY 薬品ID
  IN((RDAREA01)'薬品10',(RDAREA02));             …4
```

〔説明〕

1. プラグインモジュールで提供されたデータ型を指定します。
2. 薬品管理表中の LOB 列（SGMLTEXT）をユーザ LOB 用 RD エリア LOBAREA01 及び LOBAREA02 に分割して格納します。
3. プラグインオプションを指定します。指定方法については，プラグインのマニュアルを参照してください。
4. 薬品管理表の LOB 列構成基表をユーザ用 RD エリア RDAREA01，RDAREA02 に分割して格納します。

**(b) フレキシブルハッシュ分割，FIX ハッシュ分割の場合**

```
CREATE TABLE 薬品管理表(
薬品ID MCHAR(15),
取扱説明書 SGMLTEXT                …1
  ALLOCATE(SGMLTEXT IN((LOBAREA01),(LOBAREA02)))  …2
  PLUGIN'<DTD>medicine.dtd</DTD>'        …3
  )[FIX]※ HASH HASH6 BY 薬品ID
  IN(RDAREA01,RDAREA02)                  …4
```

注※　FIX ハッシュ分割の場合に指定します。

〔説明〕

1. プラグインモジュールで提供されたデータ型を指定します。
2. 薬品管理表中の LOB 列 SGMLTEXT をユーザ LOB 用 RD エリア LOBAREA01 及び LOBAREA02 に分割して格納します。
3. プラグインオプションを指定します。指定方法については，プラグインのマニュアルを参照してください。
4. 薬品管理表の LOB 列構成基表をユーザ用 RD エリア RDAREA01，RDAREA02 に分割して格納します。

## (2) プラグインインデクスの定義

プラグインによって提供されたデータ検索用のインデクス型をインデクスに定義するとデータを高速に検索できます。プラグインから提供されたインデクス型を定義したインデクスを**プラグインインデクス**といいます。ここでは，HiRDB Text Search Plug-in が提供するインデクス型（NGRAM）を使用したプラグインインデクスの定義方法について説明します。

CREATE INDEX で薬品管理表にプラグインインデクスを定義します。定義例を次に示します。

```
CREATE INDEX PLGINDX1
        USING TYPE NGRAM
        ON 薬品管理表(取扱説明書)
        IN ((LOBAREA03),(LOBAREA04));
```

〔説明〕

プラグインインデクス PLGINDX1 を横分割した薬品管理表に対応させて，ユーザ LOB 用 RD エリア LOBAREA03 及び LOBAREA04 に分割して格納します。なお，プラグインインデクス PLGINDX1 を構成する列に取扱説明書列を指定しています。

## (3) 表へのデータの格納

データベース作成ユティリティ（pdload）で表にデータを格納します。格納手順を次に示します。

**〈手順〉**

1. pdhold コマンドで，データロード対象 RD エリア（RDAREA01 ～ RDAREA02，LOBAREA01 ～ LOBAREA04）を閉塞します。

2. pdload コマンドで，入力データファイルを表にデータロードします。RD エリアにはデータロード対象表（インデクス）だけを格納していて，かつ初期ロードのため，データベースの更新ログ取得方式にログレスモードを選択します。また，インデクスの作成方法にインデクス一括作成モード（省略値）を選択します。また，コンストラクタ関数やコンストラクタ関数に渡すデータ型の情報を列構成情報ファイルに指定します。pdload コマンドに指定するオプションについては，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。
3. ログレスモードで pdload コマンドを実行しているため，データロード対象 RD エリアのバックアップを取得します。RD エリア単位のバックアップの取得方法については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。
4. pdrels コマンドで，データロード対象 RD エリアの閉塞を解除します。

上記のコマンドとユティリティの詳細及びこれらのコマンドとユティリティの実行結果の確認方法については，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

**補足事項**

- ログレスモードで pdload コマンドを実行するため，前記の手順 1 ～ 3 の間はデータロード対象 RD エリアを閉塞したままにしてください。
- 改竄防止表に対して pdload コマンドでデータロードするとき，-d オプションは使用できません。
- インデクス一括作成中にエラーが発生した場合の対処方法については，「6.6　インデクス一括作成中に発生したエラーの対処方法」を参照してください。

### (4) データの格納状態の確認

データロードをした場合は，運用を開始する前にデータベース状態解析ユティリティ（pddbst）を実行して，データの格納状態を確認することをお勧めします。設計どおりにデータベースを作成できたかどうかを確認できます。データベース状態解析ユティリティ（pddbst）を実行すると，次に示す情報を取得できます。

- ユーザ用 RD エリア及びユーザ LOB 用 RD エリア単位（物理解析だけ）のデータの格納状態
- レジストリ用 RD エリア及びレジストリ LOB 用 RD エリア単位（物理解析，論理解析）のデータの格納状態

### (5) ハッシュ関数で分割条件を指定している表に RD エリア単位でデータロードする場合

表分割ハッシュ関数を使用した UAP を作成し，RD エリア単位に入力データファイルを作成できます。これによって，各 RD エリアに格納されるデータ量が確認できるため，均等に分割できるハッシュ関数を選択できます。表分割ハッシュ関数を使用した UAP の作成方法については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

## 6.4.2 XML 型

ここでは，HiRDB XML Extension が提供する抽象データ型（XML 型）を定義した表の作成方法について説明します。

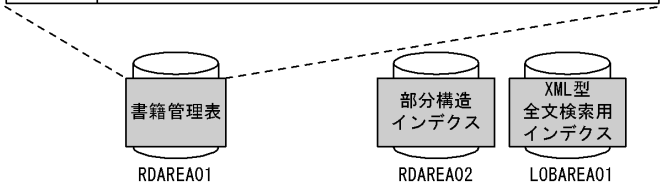
ここでは，書籍管理表を作成します。書籍管理表の作成条件を次に示します。

- ユーザ用 RD エリア RDAREA01 に書籍管理表を格納します。
- ユーザ用 RD エリア RDAREA02 に部分構造インデクスを格納します。
- ユーザ LOB 用 RD エリア LOBAREA01 に XML 型全文検索用インデクスを格納します。
- RDAREA01，RDAREA02，LOBAREA01 にはデータロード対象表（インデクス）だけを格納していて，初期ロードとします。

- データロードをするときにインデクスを一括作成（省略値）します。
- ログレスモードでデータロードをします。

書籍管理表

| 書籍ID | 書籍情報 |
|---|---|
| 452469630 | <書籍情報 書籍ID="452469630"><br><カテゴリ>データベース</カテゴリ><br><タイトル>リレーショナルデータベース解説</タイトル><br><説明>リレーショナルモデルの概念から，それに基づくRDBMSの仕組みを解説している。</説明><br></書籍情報> |
| 126513592 | <書籍情報 書籍ID="126513592"><br><カテゴリ>データベース</カテゴリ><br><タイトル>SQL徹底解説</タイトル><br><説明>最新の標準SQL言語について詳細に解説している。</説明><br></書籍情報> |
| 940123531 | <書籍情報 書籍ID="940123531"><br><カテゴリ>ネットワーク</カテゴリ><br><タイトル>図解TCP/IP</タイトル><br><説明>TCP/IPプロトコルについて図を用いながら分かりやすく説明している。</説明><br></書籍情報> |
| ⋮ | ⋮ |




## (1) 書籍管理表の定義

CREATE TABLE で書籍管理表を定義します。定義例を次に示します。

**XML 型の列を含む表の定義例**

```
CREATE TABLE 書籍管理表 (書籍ID INTEGER, 書籍情報 XML) IN RDAREA01
```

## (2) インデクスの定義

### (a) 部分構造インデクス（B-tree）

XML 型の列には，特定の部分構造をキーとし，その値をキー値としたインデクスを定義できます。このインデクスを利用すると，XMLEXISTS 述語や XMLQUERY 関数の XQuery 式中に，部分構造インデクスを定義した構造に対する述語を指定した場合，行の絞り込みの処理時間を削減できることがあります。

部分構造インデクスを利用できる XQuery 式中の述語を次に示します。

- キーである部分構造に対する比較式（=，!=，>，>=，<，<=，<>，eq，ne，lt，le，gt，ge）
- キーである部分構造に対する fn:contains 関数，fn:starts-with 関数，fn:ends-with 関数

インデクスの使用条件については，「(4) インデクスの使用条件」で説明します。

部分構造インデクスを使用した検索については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

#### (b) XML 型全文検索用インデクス（n-gram）

XML 型の列には，XML 型の値に対する全文検索用の n-gram インデクス（IXXML）を定義できます。XML 型全文検索用インデクスを定義することで，XMLEXISTS 述語の XQuery 式中で文字列一致などの全文検索条件を含む述語を記述した場合に，行の絞り込みの処理時間を削減できることがあります。

XML 型全文検索の条件となる XQuery 式中の述語を次に示します。

- 文字列（xs:string 型）同士の完全一致（=）
- fn:contains 関数
- fn:starts-with 関数
- fn:ends-with 関数
- hi-fn:contains 関数

インデクスの使用条件については，「(4) インデクスの使用条件」で説明します。

XML 型全文検索用インデクスを使用した検索については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

### (3) 表へのデータの格納

表へデータを格納する場合の入力データには，次の二つがあります。入力データの種類によって，データの格納方法が異なります。

1. XML 文書を XML 挿入データに変換した ESIS-B 形式
   この場合は，XML 変換コマンド（phdxmlcnv）又は XML 変換ライブラリ（Java ライブラリ）を使用して XML 文書を解析し，XML 挿入データ（ESIS-B 形式といいます）を生成します。この ESIS-B 形式のデータをバイナリ形式にして出力し，pdload 又は INSERT 文を使用して表へ格納します。
   XML 変換コマンド及び XML 変換ライブラリについては，マニュアル「HiRDB XML Extension Version 8」を参照してください。
2. XML 文書
   この場合は，データベース作成ユティリティ（pdload）又は XMLPARSE 関数を使用して XML 文書を XML 挿入データ（ESIS-B 形式）に変換し，表へ格納します。XML 文書から ESIS-B 形式への変換は pdload 又は XMLPARSE 関数の中で行われます。pdload 内で ESIS-B 形式への変換を行う場合には，-G オプションを指定します。

#### (a) データロード

データベース作成ユティリティ（pdload）で表にデータを格納する場合の手順を次に示します。

**〈手順〉**

1. pdhold コマンドで，データロード対象 RD エリア（RDAREA01，RDAREA02，LOBAREA01）を閉塞します。
2. pdload コマンドで，入力データファイルを表にデータロードします。
   - XML 文書をそのまま入力データとする場合は -G オプションを指定します。
   - RD エリアにはデータロード対象表（インデクス）だけを格納していて，かつ初期ロードのため，データベースの更新ログ取得方式にログレスモードを選択します。
   - インデクスの作成方法にインデクス一括作成モード（省略値）を選択します。
   - コンストラクタ関数やコンストラクタ関数に渡すデータ型の情報を列構成情報ファイルに指定します。
   - 入力データファイルの形式をバイナリ形式に設定します。

pdload コマンドに指定するオプションについては，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

3. ログレスモードで pdload コマンドを実行しているため，データロード対象 RD エリアのバックアップを取得します。RD エリア単位のバックアップの取得方法については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。
4. pdrels コマンドで，データロード対象 RD エリアの閉塞を解除します。

上記のコマンドとユティリティの詳細及びこれらのコマンドとユティリティの実行結果の確認方法については，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

**補足事項**

- ログレスモードで pdload コマンドを実行するため，前記の手順 1 ～ 3 の間はデータロード対象 RD エリアを閉塞したままにしてください。
- 改竄防止表に対して pdload コマンドでデータロードするとき，-d オプションは使用できません。
- インデクス一括作成中にエラーが発生した場合の対処方法については，「6.6　インデクス一括作成中に発生したエラーの対処方法」を参照してください。

#### (b) XML 文書の挿入

**ESIS-B 形式のデータを表へ挿入又は更新する場合**

INSERT 文の挿入値，又は UPDATE 文の更新値に XML コンストラクタ関数を指定し，その引数に生成した ESIS-B 形式のデータを設定します。
書籍管理表に埋込み変数 bookinfo に格納された XML 文書（ESIS-B 形式）の値を挿入する例を次に示します。

XML 文書（ESIS-B 形式）の挿入例

```
INSERT INTO 書籍管理表
  VALUES ( 310494321, XML(:bookinfo AS BINARY(102400)))
```

**XML 文書を表へ挿入又は更新する場合**

INSERT 文の挿入値，又は UPDATE 文の更新値に XMLPARSE 関数を指定し，その引数に XML 文書を設定します。
書籍管理表に埋込み変数 bookdoc に格納された XML 文書の値を挿入する例を次に示します。

XML 文書の挿入例

```
INSERT INTO 書籍管理表
  VALUES ( 310494321, XMLPARSE(DOCUMENT :bookdoc AS BINARY(32000)))
```

### (4) インデクスの使用条件

ここでは，(2) で説明した二つのインデクスの使用条件について説明します。

**部分構造インデクスの使用条件**

部分構造インデクスを定義した場合，次の表に示す部分構造インデクスの使用条件を満たすと，インデクスが使用されます。

**表 6-3 部分構造インデクスの使用条件**

| USING UNIQUE TAG の指定 | XQuery 指定箇所 | XQuery 中の演算子又は関数 | 部分構造インデクスの使用条件※ |
|---|---|---|---|
| あり | XMLEXISTS 述語 | = | (a)1，2，3<br>(b)1，2，5，6，7，8，9 |
| | | !=，>，>=，<，<=，<>，eq，ne，gt，ge，lt，le，<br>fn:contains，<br>fn:starts-wtih，<br>fn:ends-with | (a)1，2，3<br>(b)1，2，4，6，7，8，9 |
| | XMLQUERY 関数 | = | (c)1，2，3 |
| なし | XMLEXISTS 述語 | = | (a)1，2，3<br>(b)1，3，5，6，7，8，9 |
| | | !=，>，>=，<，<=，<> | (a)1，2，3<br>(b)1，3，4，6，7，8，9 |
| | | fn:contains<br>fn:starts-wtih<br>fn:ends-with | (a)1，2，3<br>(b)1，3，4，6，7，8，10 |
| | XMLQUERY 関数 | = | (c)1，2，3 |

注※

(a) は，(a) で説明する部分構造インデクス及び XML 型全文検索用インデクス共通の使用条件を示します。

(b) は，(b) で説明する部分構造インデクスの使用条件を示します。

(c) は，(c) で説明する XMLQUERY 関数の XQuery に関する部分構造インデクスの使用条件を示します。

また，数字は (a)，(b)，及び (c) での項番に対応しています。

### XML 型全文検索用インデクスの使用条件

XML 型全文検索用インデクスを定義した場合，次の表に示す XML 型全文検索用インデクスの使用条件を満たすと，インデクスが使用されます。

**表 6-4 XML 型全文検索用インデクスの使用条件**

| XQuery 指定箇所 | XQuery 中の演算子又は関数 | XML 型全文検索用インデクスの使用条件※ |
|---|---|---|
| XMLEXISTS 述語 | =<br>fn:contains<br>fn:starts-wtih<br>fn:ends-with | (a)1，2，3<br>(d)1，2，4，5，6，7，8 |
| | hi-fn:contains | (a)1，2，3<br>(d)1，3，4，5，6，7，8 |

注※

(a) は，(a) で説明する部分構造インデクス及び XML 型全文検索用インデクス共通の使用条件を示します。

(d) は，(d) で説明する XML 型全文検索用インデクスの使用条件を示します。

また，数字は (a) 及び (d) での項番に対応しています。

なお，HiRDB が見積もるアクセスコストによっては，これらのインデクスを使用しないことがあります。インデクスを使用した検索が行われたかどうかについては，アクセスパス表示ユティリティ（pdvwopt）で確認してください。

また，XMLEXISTS 述語の XQuery 問合せ中に指定した，部分構造インデクス，又は XML 型全文検索インデクスを使用できる述語については，最大で 255 個しかインデクスを用いて評価しません。

部分構造インデクス，及び XML 全文検索用インデクスの両方が使用可能な XQuery 中の演算子，又は関数を使用して検索する場合，使用されるインデクスは演算子，又は関数によって決まります。異なるインデクスを使用したい場合は，マニュアル「HiRDB Version 8 SQL リファレンス」の「使用インデクスの SQL 最適化指定」で使用するインデクスを指定することができます。

**表 6-5　複数のインデクスが使用可能な演算子又は関数の場合に使用されるインデクス**

| 部分構造インデクス，及び XML 型全文検索用インデクスの両方が使用可能な XQuery 中の演算子又は関数 | 検索に使用されるインデクス |
|---|---|
| = | 部分構造インデクス |
| fn:contains | XML 型全文検索用インデクス |
| fn:starts-wtih | 部分構造インデクス |
| fn:ends-with | XML 型全文検索用インデクス |

### (a) 部分構造インデクス及び XML 型全文検索用インデクス共通の使用条件

部分構造インデクス及び XML 型全文検索用インデクス共通の使用条件を次に示します。

なお，次のように定義したインデクスを例文中で使用します。

```
create index idx1 on t1(c1) key using unique tag from '/root/elm1/@attr1' as
varchar(10)
```

1. XMLEXISTS 述語の XML 問合せ引数に XML 問合せ文脈項目を指定している。

   （例）

   ```
   select c2 from t1
     where xmlexists('/root/elm1[@attr1 eq "ABC"]'
           passing by value c1,'DEF' as A)
   ```

   注　アンダーライン部分が XML 問合せ文脈項目です。

2. XMLEXISTS 述語の XQuery 問合せ中に指定した文脈項目式（ピリオド）をすべて XQuery 述語中に指定している。

   （例）

   ```
   select c2 from t1
     where xmlexists('/root/elm1[./@attr1 eq "ABC"]' passing by value c1)
   ```

   注　アンダーライン部分が XQuery 述語中に指定した文脈項目式です。

3. XMLEXISTS 述語の XQuery 問合せ中の XQuery 論理式（AND，OR）をすべて XQuery 述語中に指定している。

   （例）

   ```
   select c2 from t1
     where xmlexists('/root[elm1/@attr1 = "ABC" or elm1/@attr1 = "DEF"]'
           passing by value c1)
   ```

   注　アンダーライン部分が XQuery 述語中に指定した XQuery 論理式です（OR）。

### (b) 部分構造インデクスの使用条件

部分構造インデクスの使用条件を次に示します。

なお，1.，2.，4. ～ 9. では，次のように定義したインデクスを例文中で使用します。

```
  create index idx1 on t1(c1) key using unique tag from '/root/elm1/@attr1' as
varchar(10)
  create index idx4 on t1(c1) key using unique tag from '/root/elm1/elm2
' as varchar(10)
```

3.，10. では，次のように定義したインデクスを例文中で使用します。

```
create index idx2 on t1(c1) key from '/root/elm1/@attr1' as varchar(10)
create index idx5 on t1(c1) key from '/root/elm1/elm2' as varchar(10)
```

1. **インデクスの部分構造指定と，**XMLEXISTS **述語の** XQuery **問合せ中で条件として指定した部分構造パスが一致する。**
   （例）
   ```
   select c2 from t1
     where xmlexists('/root/elm1[@attr1 eq "ABC"]' passing by value c1)
   ```
   注　アンダーライン部分が一致する部分構造パスです。

2. USING UNIQUE TAG **の指定がある部分構造インデクスの場合，汎用比較，値比較，**fn:contains **関数，**fn:starts-with **関数，又は** fn:ends-with **関数を使用して，**XMLEXISTS **述語の** XQuery **問合せ中で条件として指定した部分構造パスを比較する。**
   （例）
   ```
   select c2 from t1
     where xmlexists('/root/elm1[@attr1 eq "ABC"]' passing by value c1)
   ```
   注　アンダーライン部分が XQuery 比較式です（値比較）。

3. USING UNIQUE TAG **の指定がない部分構造インデクスの場合，汎用比較，**fn:contains **関数，**fn:starts-with **関数，又は** fn:ends-with **関数を使用して，**XMLEXISTS **述語の** XQuery **問合せ中で条件として指定した部分構造パスを比較する。**
   （例）
   ```
   select c2 from t1
     where xmlexists('/root/elm1[@attr1 = "ABC"]' passing by value c1)
   ```
   注　アンダーライン部分が XQuery 比較式です（汎用比較）。

4. **汎用比較又は値比較と，**fn:contains **関数，**fn:starts-with **関数，又は** fn:ends-with **関数の場合に分けて説明します。**
   **＜汎用比較，又は値比較の場合＞**
   汎用比較，又は値比較で比較する対象は，XMLEXISTS 述語の XQuery 問合せ中で条件として指定した部分構造パスと，一つの XQuery 定数又は XQuery 変数である。
   （例）
   ```
   select c2 from t1
     where xmlexists('/root/elm1[@attr1 >= "ABC"]' passing by value c1)
   ```
   注　アンダーライン部分が部分構造パスと XQuery 定数の比較です。

   **＜** fn:contains **関数，**fn:starts-with **関数，又は** fn:ends-with **関数の場合＞**
   fn:contains 関数，fn:starts-with 関数，又は fn:ends-with 関数の第 1 引数が XMLEXISTS 述語の XQuery 問合せ中で条件として指定した部分構造パスで，かつ第 2 引数が一つの XQuery 定数又は XQuery 変数である。
   （例）
   ```
   select c2 from t1
     where xmlexists('/root/elm1[starts-with(@attr1,"ABC")]' passing by value c1)
   ```
   注　アンダーライン部分が部分構造パスと XQuery 定数の比較です。

5. = **で比較する対象は，**XMLEXISTS **述語の** XQuery **問合せ中で条件として指定した部分構造パスと，システム共通定義** pd_apply_search_ats_num **オペランドの指定値以下の** XQuery **定数又は** XQuery **変数から成る** XQuery **シーケンス連結式である。**
   （例）

```
select c2 from t1
  where xmlexists('/root/elm1[@attr1 = ("ABC","DEF","GHI")]'
        passing by value c1)
```

注　アンダーライン部分がシステム共通定義 pd_apply_search_ats_num オペランドの指定値以下の XQuery 定数から成る XQuery シーケンス連結式です。

6. **部分構造インデクス定義時に指定したキー値のデータ型が，**XMLEXISTS **述語の** XQuery **問合せ中で条件として指定した部分構造パスと比較する** XQuery **定数若しくは** XQuery **問合せ中の** XQuery **変数に渡す値式のデータ型と同じ，又は変換可能である。**

（例）

```
select c2 from t1
  where xmlexists('/root/elm1[@attr1 = "ABC"]' passing by value c1)
```

注　アンダーライン部分が，キー値のデータ型である VARCHAR 型と同じ string 型データです。

7. XMLEXISTS **述語の** XQuery **問合せに指定した** XQuery **論理式（**OR**）オペランドに部分構造インデクスを使用できる条件だけを含む。**

（例）

```
select c2 from t1
  where xmlexists('/root[elm1/@attr1 = "ABC" or elm1/@attr1 = "DEF"]'
        passing by value c1)
```

注　アンダーライン部分が，すべて部分構造インデクスを使用できる条件です。

8. 4. **又は** 5. **で，条件として指定した部分構造パスと比較する値として** XQuery **変数を指定した場合，その** XQuery **変数に渡す値式が定数，**USER**，又は外への参照なしスカラ副問合せのどれかである。**

（例）

```
select c2 from t1
  where xmlexists('/root/elm1[@attr1 eq $A]'
        passing by value c1,'ABC' as A)
```

注　アンダーライン部分が XQuery 問合せ中の XQuery 変数に渡す値式です（定数）。

9. **値比較，又は汎用比較で比較する対象の部分構造パスが次の形式で指定されている。又は，**USING UNIQUE TAG **の指定がある部分構造インデクスの場合で，かつ** fn:contains **関数，**fn:starts-with **関数，又は** fn:ends-with **関数の第** 1 **引数を含む部分構造パスが次の形式で指定されている。**

```
部分構造パス:: = 〔XML名前空間宣言〕…部分構造パス式
XML名前空間宣言::= {declare namespace 接頭辞 = XML名前空間URI;
                    | declare default element namespace XML名前空間URI;}
部分構造パス式:: = 〔/ステップ式…〕/ステップ式
ステップ式:: = {〔{child:: | attribute:: | @ }〕 修飾名 | 文脈項目式}
文脈項目式:: = ピリオド
ピリオド::= .
修飾名:: = 〔接頭辞:〕局所名
```

以下，（例 1）～（例 3）は値比較，又は汎用比較を使用した例です。下記の例で，値比較 eq に他の値比較又は汎用比較を指定した場合も，インデクスが使用されます。
USING UNIQUE TAG の指定有無に関連するインデクスの使用有無は 2. と 3. を参照してください。

（例 1）ステップ式に @，又は attribute:: と，修飾名を指定した場合（@ と attribute:: は同義）

```
select c2 from t1
  where xmlexists('/root/child::elm1[@attr1 eq "ABC"]'
        passing by value c1)
```

注　アンダーライン部分が上記の形式と一致する部分構造パスです。

（例 2）ステップ式に child:: を指定，又は child:: を省略し，修飾名を指定した場合

```
select c2 from t1
  where xmlexists('/root/child::elm1[child::elm2 eq "ABC"]'
```

```
        passing by value c1)
```

注　アンダーライン部分が上記の形式と一致する部分構造パスです。

（例 3）ステップ式に文脈項目式を指定した場合（値比較は 2. に該当する場合だけ）

```
select c2 from t1
  where xmlexists('/root/child::elm1/elm2[ . eq "ABC"]'
        passing by value c1)
```

注　アンダーライン部分が上記の形式と一致する部分構造パスです。

以下，（例 4）～（例 6）は XQuery 関数を使用した例です。下記の例で，fn:starts-with に他の XQuery 関数を指定した場合も，インデクスが使用されます。

（例 4）ステップ式に @，又は attribute:: と，修飾名を指定した場合（@ と attribute:: は同義）

```
select c2 from t1
  where xmlexists('/root/elm1[fn:starts-with(@attr1 ,"ABC")]'
        passing by value c1)
```

注　アンダーライン部分が上記の形式と一致する部分構造パスです。

（例 5）ステップ式に child:: を指定，又は child:: を省略し，修飾名を指定した場合

```
select c2 from t1
  where xmlexists('/root/elm1[fn:starts-with(elm2 ,"ABC")]'
        passing by value c1)
```

注　アンダーライン部分が上記の形式と一致する部分構造パスです。

（例 6）ステップ式に文脈項目式を指定した場合

```
select c2 from t1
  where xmlexists('/root/elm1/@attr1[fn:starts-with( . ,"ABC")]'
        passing by value c1)
```

注　アンダーライン部分が上記の形式と一致する部分構造パスです。

10. **USING UNIQUE TAG の指定がない部分構造インデクスの場合で，かつ** fn:contains **関数，**fn:starts-with **関数，又は** fn:ends-with **関数の第** 1 **引数を含む部分構造パスが次の形式で指定されており，さらに第** 1 **引数が次のステップ式終端の形式で指定されている。**

```
部分構造パス:: = 〔XML名前空間宣言〕…部分構造パス式
XML名前空間宣言::= {declare namespace 接頭辞 = XML名前空間URI;
                    | declare default element namespace XML名前空間URI;}
部分構造パス式:: = 〔/ステップ式…〕/ステップ式終端
ステップ式:: = {〔{child:: | attribute:: | @ }〕 修飾名 | 文脈項目式}
ステップ式終端:: = {{attribute:: | @ } 修飾名 | 文脈項目式}
文脈項目式:: = ピリオド
ピリオド::= .
修飾名:: = 〔接頭辞:〕局所名
```

下記の例で，fn:starts-with に他の XQuery 関数を指定した場合も，インデクスを使用します。

（例 1）ステップ式終端に @，又は attribute:: と，修飾名を指定した場合（@ と attribute:: は同義）

```
select c2 from t1
  where xmlexists('/root/elm1[fn:starts-with(@attr1 ,"ABC")]'
        passing by value c1)
```

注　アンダーライン部分が上記の形式と一致する部分構造パスです。

（例 2）ステップ式終端に文脈項目式を指定した場合

```
select c2 from t1
  where xmlexists('/root/elm1/@attr1[fn:starts-with( . ,"ABC")]'
        passing by value c1)
```

注　アンダーライン部分が上記の形式と一致する部分構造パスです。

### (c) XMLQUERY 関数の XQuery に関する部分構造インデクスの使用条件

XMLQUERY 関数の XQuery に関する部分構造インデクスの使用条件を次に示します。

なお，1. 及び 2. では，次のように定義したインデクスを例文中で使用します。

```
  create index idx1 on t1(c1) key using unique tag from '/root/elm1' as
varchar(10)
```

1. SQL が次のすべての条件を満たす。
   - SELECT 文，又は INSERT ～ SELECT 文である
   - 主問合せの選択式が一つである
   - 上記の主問合せの選択式が XMLQUERY である（ただし，XMLSERIALIZE の引数が XMLQUERY でもよい）
   - 上記の XMLQUERY の XML 問合せ引数が XML 問合せ変数一つで，かつ変数に渡す値式は XMLAGG である
   - 上記の XML 問合せ変数に指定した XMLAGG の引数は列指定単独である
   - 表の結合を指定していない
   - 集合演算を指定していない
   - 副問合せを指定していない
   - 集合関数を指定していない
   - GROUP BY 句を指定していない
   - HAVING 句を指定していない
   - WHERE 句が指定されている場合，AND の指定数が 255 以下である

   SQL の例については，2. の例を参照してください。

2. XMLQUERY 関数に指定した XQuery が次のすべての条件を満たす。
   a. XML 問合せ変数をルートとするパス式である
   b. 最も外側の XQuery 述語の指定が一つだけ存在する
   c. b. の述語で汎用比較 '=' による比較を行っている
   d. c. の比較が，XML 列の特定の部分構造と XQuery 変数をルートとするパス式の比較である
   （例）

```
select xmlserialize(xmlquery('$VAR1/root[elm1 = $VAR1/root[elm2 = "ABC"]/
elm1]/elm1'
  passing by value xmlagg(c1) as VAR1 empty on empty) as varchar(32000)) from t1
```

   注　アンダーライン部分が上記の条件と一致する箇所です。

3. 2. の d. の XML 列の特定の部分構造，及び XQuery 変数をルートとするパス式に，同じデータ型の部分構造インデクスを定義している（これらのインデクスは同一のものでもかまいません）。

### (d) XML 型全文検索用インデクスの使用条件

XML 型全文検索用インデクスの使用条件を次に示します。

1. 検索対象の XML 型の列に全文検索インデクスを定義している。
   （例）

```
create index idx3 using type ixxml on t2(c1) in (LOB1)
```

2. fn:contains 関数，fn:starts-with 関数，又は fn:ends-with 関数の第 1 引数を含む部分構造パスが次の形式で指定されており，さらに第 1 引数が次のテキストステップ式終端，又は属性ステップ式終端の形式で指定されている。又は = で比較する対象の部分構造パスが次の形式で指定されており，XQuery 述語中に指定した部分構造パスが次のテキストステップ式終端又は属性ステップ式終端の形式で指定されている。

```
部分構造パス:: = 〔XML名前空間宣言〕… 部分構造パス式
XML名前空間宣言:: = {declare default element namespace
                      "http://www.w3.org/XML/1998/namespace";
                    |declare namespace 接頭辞 = XML名前空間URI;※
                    |declare default element namespace XML名前空間URI;※}
部分構造パス式:: = 〔{/ | //※} ステップ式 …〕
                       {/ | //※} {テキストステップ式 | 属性ステップ式終端}
ステップ式:: = {〔child::〕 名前テスト | 文脈項目式}
テキストステップ式:: = 〔{child::|descendant::}〕テキストテスト
                              /テキストステップ式終端
テキストステップ式終端:: = 文脈項目式
属性ステップ式終端:: = {{attribute:: | @} 名前テスト
                      | 〔{attribute:: | @}〕 属性テスト}
文脈項目式:: = ピリオド
ピリオド:: = .
名前テスト:: = {修飾名|*※|接頭辞:*※|*:局所名※}
修飾名:: = 〔接頭辞:〕局所名
```

注※　HiRDB XML Extension のバージョンが 08-04 以降の場合に指定できます。

下記の例で，fn:contains に他の XQuery 関数を指定した場合も，インデクスを使用します。

（例 1）第一引数に @，又は attribute:: と，名前テストを指定した場合（@ と attribute:: は同義）

```
select c2 from t2
  where xmlexists('/root/child::elm1[fn:contains(@attr1,"ABC")]'
        passing by value c1)
```

注　アンダーライン部分が上記の形式と一致する部分構造パスです。

（例 2）第一引数に @，又は attribute:: と，属性テストを指定した場合（@ と attribute:: は同義）

```
select c2 from t2
  where xmlexists('/root/child::elm1[fn:contains(@attribute(),"ABC")]'
        passing by value c1)
```

注　アンダーライン部分が上記の形式と一致する部分構造パスです。

（例 3）第一引数に属性テストのみを指定した場合

```
select c2 from t2
  where xmlexists('/root/child::elm1[fn:contains(attribute(),"ABC")]'
        passing by value c1)
```

注　アンダーライン部分が上記の形式と一致する部分構造パスです。

（例 4）第一引数に文脈項目式を指定した場合

```
select c2 from t2
  where xmlexists('/root/child::elm1/text()[fn:contains( . ,"ABC")]'
        passing by value c1)
```

注　アンダーライン部分が上記の形式と一致する部分構造パスです。

3. hi-fn:contains **関数の第** 1 **引数を含む部分構造パスが次の形式で指定されており，さらに第** 1 **引数が次のテキストステップ式又は属性ステップ式終端の形式で指定されている。また，**HiRDB XML Extension **のバージョンが** 08-04 **以降である。**

```
部分構造パス:: = 〔XML名前空間宣言〕… 部分構造パス式
XML名前空間宣言:: = {declare namespace 接頭辞 = XML名前空間URI;
                    |declare default element namespace XML名前空間URI;}
部分構造パス式:: = 〔{/ | //} ステップ式 …〕
                       {/ | //} {テキストステップ式 | 属性ステップ式終端}
ステップ式:: = {〔child::〕 名前テスト | 文脈項目式}
テキストステップ式:: = 〔{child::|descendant::}〕テキストテスト
                              〔/テキストステップ式終端〕
テキストステップ式終端:: = 文脈項目式
属性ステップ式終端:: = {{attribute:: | @} 名前テスト
```

```
                    | 〔{attribute:: | @}〕 属性テスト}
文脈項目式:: = ピリオド
ピリオド:: = .
名前テスト:: = {修飾名 | * | 接頭辞:* | *:局所名}
修飾名:: = 〔接頭辞:〕局所名
```

（例 1）第一引数にテキストを指定した場合

```
select c2 from t2
  where xmlexists('/root/elm1[hi-fn:contains(text(),"""ABC AND DEF""")]'
        passing by value c1)
```

注　アンダーライン部分が上記の形式と一致する部分構造パスです。

```
select c2 from t2
  where xmlexists('/root/elm1[hi-fn:contains(descendant::text(),"""ABC AND
DEF""")]'
        passing by value c1)
```

注　アンダーライン部分が上記の形式と一致する部分構造パスです。

（例 2）第一引数にテキストテストおよび文脈項目式を指定した場合

```
select c2 from t2
  where xmlexists('/root/elm1[hi-fn:contains(text()/.,"""ABC AND DEF""")]'
        passing by value c1)
```

注　アンダーライン部分が上記の形式と一致する部分構造パスです。

```
select c2 from t2
  where xmlexists('/root/elm1[hi-fn:contains(descendant::text()/.,"""ABC AND
DEF""")]'
        passing by value c1)
```

注　アンダーライン部分が上記の形式と一致する部分構造パスです。

（例 3）第一引数に文脈項目式を指定した場合

```
select c2 from t2
  where xmlexists('/root/elm1/text()[hi-fn:contains( . ,"""ABC AND DEF""")]'
        passing by value c1)
```

注　アンダーライン部分が上記の形式と一致する部分構造パスです。

```
select c2 from t2
  where xmlexists('/root/elm1/ descendant::text()[hi-fn:contains( . ,"""ABC
AND DEF""")]'
        passing by value c1)
```

注　アンダーライン部分が上記の形式と一致する部分構造パスです。

（例 4）第一引数に @ 又は attribute:: と，名前テストを指定した場合（@ と attribute:: は同義）

```
select c2 from t2
  where xmlexists('/root/elm1[hi-fn:contains(@attr1,"""ABC AND DEF"
""")]'
        passing by value c1)
```

注　アンダーライン部分が上記の形式と一致する部分構造パスです。

（例 5）第一引数に @，又は attribute:: と，属性テストを指定した場合（@ と attribute:: は同義）

```
select c2 from t2
  where xmlexists('/root/elm1[hi-fn:contains(@attribute(),"""ABC AND DEF""")]'
        passing by value c1)
```

注　アンダーライン部分が上記の形式と一致する部分構造パスです。

（例 6）第一引数に属性テストのみを指定した場合

```
select c2 from t2
  where xmlexists('/root/elm1[hi-fn:contains(attribute(),"""ABC AND DEF")]'
        passing by value c1)
```

注　アンダーライン部分が上記の形式と一致する部分構造パスです。

4. XMLEXISTS 述語の XQuery 問合せ中に指定した文字列の長さが 32,000 バイト以下である。

（例）

```
select c2 from t2
  where xmlexists('/root/elm1[fn:contains(@attr1,"ABCDEF")]'
        passing by value c1)
```

注　アンダーライン部分が 32,000 バイト以下の文字列です。

5. 2. 又は 3. の形式で指定した部分構造パスと比較する値が文字列の XQuery 定数である。

（例）

```
select c2 from t2
  where xmlexists('/root/child::elm1[fn:contains(@attr1,"ABC")]'
        passing by value c1)
```

注　アンダーライン部分が文字列の XQuery 定数です。

6. XMLEXISTS 述語の XQuery 問合せ中の部分構造パス式に指定した，/，//，@，及び局所名の長さの和（部分構造パス式の 1 文字目の / は除く）が 1,024 バイト以下である。

（例）

```
select c2 from t2
  where xmlexists('/root/elm1[fn:contains(@attr1,"ABCDEF")]'
        passing by value c1)
```

注　アンダーライン部分が 1,024 バイト以下の部分構造パス式です。

7. XMLEXISTS 述語の XQuery 問合せ中に指定した部分構造パス式中での // の指定が一つ以内である。

（例）

```
select c2 from t2
  where xmlexists('/root//elm1[fn:contains(@attr1,"ABCDEF")]'
        passing by value c1)
```

注　アンダーライン部分が // の指定が一つ以内の部分構造パス式です。

8. 検索対象の XML 型の列に定義した全文検索インデクスに，以下のどのプラグインオプションも指定していない。

プラグインオプションの詳細は，マニュアル「HiRDB XML 拡張機能 HiRDB XML Extension Version 8」の「2.5.4　インデクスの定義」を参照してください。

- DELcode= ファイル名
- NOindex= ファイル名
- ENGLISH
- ENGLISH_STANDARD

（例）

```
create index idx6 using type ixxml on t2(c1) in (LOB1) PLUGIN'SAMECASE=
ON,SAMEWIDE=ON,SAMEY=ON,SAMED=ON,DELcode=ON'
```

（アンダーラインがインデクスを使用可能なオプションだけを指定した全文検索インデクス）

**(e) インデクスを使用しない場合**

次に示す 1. ～ 4. の場合は，部分構造インデクス又は XML 型全文検索用インデクスを使用しません。

なお，1.，3. では，次のように定義したインデクスを例文中で使用します。

```
create index idx1 on t1(c1) key using unique tag from '/root/elm1/@attr1' as
archar(10)
```

2.，4. では次のように定義したインデクスを例文中で使用します。

```
create index idx3 using type ixxml on t2(c1) in (LOB1)
```

1. **‘** /root[elm1/@attr1 **’ が ”** ABC **” 又は ”** DEF **” であるかを評価する** XQuery **を** XMLEXISTS **述語の** XQuery **問合せに指定する場合**

（例：変更前）

```
select c2 from t1
  where xmlexists('/root[elm1/@attr1 = "ABC"] or /root[elm1/@attr1 = "DEF"]'
        passing by value c1)
```

注　アンダーライン部分が XQuery 述語中に指定していない XQuery 論理式（OR）です。
上記の例では，XMLEXISTS 述語の XQuery 問合せ中の XQuery 論理式（AND，OR）すべてを XQuery 述語中に指定していないため，インデクスを使用しません。この指定方法では，XMLEXISTS 述語の引数の XQuery 問合せ（XQuery 述語中を除く）に直接 XQuery 論理式を指定しているため，XQuery 問合せの結果は必ず真か偽どちらかのブーリアン値になります。このため，XQuery 問合せの結果は空のシーケンスではないので XMLEXISTS 述語の結果は常に真となり，意図した結果になりません（XMLEXISTS 述語は XQuery 問合せの結果が空のシーケンスの場合だけ偽になり，それ以外は真になる）。
次のように変更すると，インデクスを使用するようになります。

（例：変更後）

```
select c2 from t1
  where xmlexists('/root[elm1/@attr1 = "ABC" or elm1/@attr1 = "DEF"]'
        passing by value c1)
```

注　アンダーライン部分が XQuery 述語中に指定した XQuery 論理式（OR）です。

2. **‘** /root/elm1 **’ 以下のテキストノードを検索する** XQuery **を** XMLEXISTS **述語の** XQuery **問合せに指定する場合**

次の例では，contains 関数の第 1 引数を含む部分構造パスのテキストステップ式終端が，(4)(c) の 2. で示した形式と一致しないため，インデクスを使用しません。

（例：変更前）

```
select c2 from t2
  where xmlexists('/root[fn:contains(elm1/text(),"ABC")]' passing by value c1)
```

注　アンダーライン部分が，(4)(c) の 2. で示した形式と一致しない部分構造パスです。
次のように変更すると，インデクスを使用するようになります。

（例：変更後）

```
select c2 from t2
  where xmlexists('/root/elm1/text()[fn:contains( . ,"ABC")]'
        passing by value c1)
```

注　アンダーライン部分が，(4)(c) の 2. で示した形式と一致する部分構造パスです。

3. **‘** /root/elm1/@attr1/ **’ が ”** ABC **” であるかを評価する** XQuery **を** XMLEXISTS **述語の** XQuery **問合せに指定する場合**

XMLEXISTS 述語の XQuery 問合せ中で条件として指定した部分構造パスの部分構造パス式が，(4)(b) の 9. で示した形式と一致しないため，インデクスを使用しません。

（例：変更前）

```
select c2 from t1
  where xmlexists('$A/root/elm1[@attr1 eq "ABC"]'
        passing by value c1,c1 as A)
```

注　アンダーライン部分が，(4)(b) の 9. で示した形式と一致しない部分構造パスです。
次のように変更すると，インデクスを使用するようになります。

（例：変更後）

```
select c2 from t1
  where xmlexists('/root/elm1[@attr1 eq "ABC"]'
        passing by value c1,c1 as A)
```

注　アンダーライン部分が，(4)(b) の 9. で示した形式と一致する部分構造パスです。

4. **論理演算子（OR，AND）で XMLEXISTS 述語を多数連結した場合**

HiRDB が見積もるアクセスコストにより，インデクスを使用しない方が最適なアクセスパスとなると判断し，XMLEXISTS 述語の評価にインデクスを使用しません。hi-fn:contains 関数を指定した場合はインデクスだけで評価できないため，SQL エラーとなります。

（例：変更前）

```
select c2 from t2
  where xmlexists('/root/elm1[hi-fn:contains(text(),"""01ABC""")]' passing by
value c1)
     or xmlexists('/root/elm1[hi-fn:contains(text(),"""02ABC""")]' passing by
value c1)
          ・・・(省略)・・・
     or xmlexists('/root/elm1[hi-fn:contains(text(),"""30ABC""")]' passing by
value c1)
```

注　アンダーライン部分が，単独の指定ではインデクスを使用する XMLEXISTS 述語を 30 個指定した条件です。

次のように変更すると，インデクスを使用するようになります。

（例：変更後）

```
select c2 from t2 with index(idx3,idx3)
  where xmlexists('/root/elm1[hi-fn:contains(text(),"""01ABC""")]' passing by
value c1)
     or xmlexists('/root/elm1[hi-fn:contains(text(),"""02ABC""")]' passing by
value c1)
          ・・・(省略)・・・
     or xmlexists('/root/elm1[hi-fn:contains(text(),"""30ABC""")]' passing by
value c1)
```

注　アンダーライン部分が，複数インデク利用の実行に必要な種類のインデクスを指定したインデクスの SQL 最適化指定です。

# 6.5 ユーザが定義した抽象データ型を定義した表の作成

## 6.5.1 抽象データ型の定義

抽象データ型とルーチンを使用して，複雑な構造を持つデータ型とその操作を独自に定義し，利用できます。

### (1) 定義方法

任意の構造を持つデータ型（抽象データ型）を定義するには，定義系 SQL の CREATE TYPE を実行します。CREATE TYPE では，データ構造とそのデータに対する操作を定義します。ここでは，次に示すデータ構造を持つ抽象データ型「t_ 従業員」を定義し，それに対する操作を関数として定義します。

**データ構造**

- 氏名，性別，役職，入社年月日，顔写真，基本給というデータ構造を定義します。

**データ操作**

- 現在の日付と入社年月日から勤続年数を算出します。
- 勤続年数に応じた報酬率を算出します。
- 基本給に報酬率を乗じて従業員の報酬を算出します。

（例）

```
CREATE TYPE t_従業員 (                    …1
   PUBLIC    氏名     NCHAR(16),
             性別     CHAR(1),
             役職     NCHAR(10),
   PRIVATE   入社年月日 date,          …2
   PUBLIC    顔写真   BLOB(64K),
   PROTECTED 基本給   INTEGER,         …3

   PUBLIC FUNCTION t_従業員(p_氏名 NCHAR(16),    …4
                  p_性別 CHAR(1),
                  p_役職 NCHAR(10),
                  p_入社年月日 date,
                  p_顔写真 BLOB(64K),
                  p_基本給 INTEGER)
            RETURNS t_従業員
       BEGIN
            DECLARE d_従業員 t_従業員;     …5
            SET d_従業員=t_従業員();       …6
            SET d_従業員..氏名=p_氏名;     …7
            SET d_従業員..性別=p_性別;     …7
            SET d_従業員..役職=p_役職;     …7
            SET d_従業員..入社年月日=p_入社年月日;    …7
            SET d_従業員..顔写真=p_顔写真;     …7
            SET d_従業員..基本給=p_基本給;     …7
            RETURN d_従業員;                  …8
       END,

   PUBLIC FUNCTION 勤続年数(p t_従業員) RETURNS INTEGER    …9
       BEGIN
            DECLARE working_years INTERVAL YEAR TO DAY;
            SET working_years=CURRENT_DATE - p..入社年月日;
            RETURN YEAR(working_years);
       END,

   PROTECTED FUNCTION 報酬率(p t_従業員) RETURNS FLOAT     …10
```

```
        BEGIN
             DECLARE rate FLOAT;
             SET rate=勤続年数(p)*0.2/30;
             RETURN rate;
        END,

    PUBLIC FUNCTION 報酬(p t_従業員) RETURNS INTEGER   …11
        BEGIN
             DECLARE bonus INTEGER;
             SET bonus=p..基本給*報酬率(p);
             RETURN bonus;
        END
  )
```

〔説明〕

1. データ構造の定義です。抽象データ型「t_ 従業員」を定義します。
2. 「t_ 従業員」型の属性「入社年月日」は，報酬査定の算出に使用します。この属性は，外部から直接参照又は変更する必要がないため，隠蔽レベル「PRIVATE」を指定しています。隠蔽レベルについては，「12.17　抽象データ型を含む表」を参照してください。
3. 「t_ 従業員」型の属性「基本給」は，報酬査定の算出に使用します。この属性も，外部から直接参照又は変更する必要がありません。ただし，この属性はサブタイプでも共通に参照するため，隠蔽レベル「PROTECTED」を指定しています。隠蔽レベルについては，「12.17　抽象データ型を含む表」を参照してください。
4. ユーザ定義のコンストラクタ関数を定義します。
5. 値（インスタンス）を生成し，関数の戻り値とするための SQL 変数を t_ 従業員型で宣言します。
6. システムが提供するデフォルトコンストラクタ関数によって，すべての属性の値がナル値である値（インスタンス）を生成します。デフォルトコンストラクタ関数は，引数なしの t_ 従業員型と同じ名称の関数です。
7. 6. で指定した値に対し，コンポネント指定による代入文で，各属性の値を代入します。コンストラクタ関数の引数から得た値を設定したり，その値を使ってデータを加工したものを代入したりできます。
8. RETURN 文によって，新しく生成した値（インスタンス）を戻り値とします。戻り値のデータ型は，t_ 従業員型でなければなりません。これは，抽象データ型名と同じ名前のコンストラクタ関数であること及び RETURNS 句で型を決めているためです。
9. データ操作のための関数です。従業員の勤続年数を返します。現在の日付と入社年月日から算出します。隠蔽レベル「PRIVATE」が指定されている属性「入社年月日」にアクセスする関数です。
10. データ操作のための関数です。従業員の報酬率を返します。勤続年数に応じて算出します。
11. データ操作のための関数です。従業員の報酬を返します。基本給に報酬率を乗じて，勤続年数に応じた従業員の報酬を算出します。

### (2) 継承を利用する場合の定義方法

抽象データ型「t_ 従業員」をスーパタイプとするサブタイプ「t_ 営業部員」の定義例を次に示します。

(例)

```
CREATE TYPE t_営業部員 UNDER t_従業員
(       PUBLIC 担当顧客       NCHAR(15),
        PUBLIC FUNCTION 報酬(p t_営業部員) RETURNS INTEGER
          BEGIN
               DECLARE salebonus INTEGER;
               SET salebonus = 顧客総数(・・・)*1000+p..基本給*報酬率(p);
               RETURN salebonus;
          END
)
```

## (3) 抽象データ型のナル値

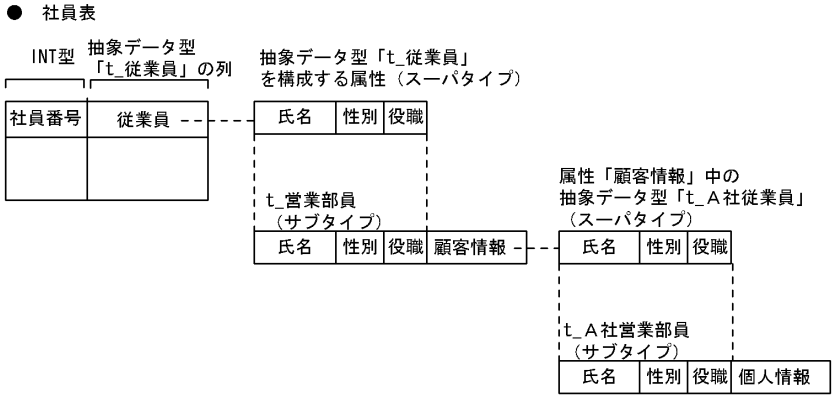
操作系 SQL の INSERT 文で値が明示的に指定されない場合，抽象データ型全体の値はナル値になります。

## (4) 抽象データ型のサブタイプを削除する方法

表定義で，ある抽象データ型を直接指定していない場合でも，その抽象データ型の上位の抽象データ型（スーパタイプ）が列の型に指定されている場合，代替可能性によってその抽象データ型（サブタイプ）の値は表に格納されている場合があります。このため，抽象データ型（サブタイプ）を削除する場合には注意が必要です。

次の図に示す**代替可能性**を利用した抽象データ型を含む表の場合にサブタイプを削除する方法について説明します。

**図 6-3　代替可能性を利用した抽象データ型を含む表の例**




1. 社員表を削除します。
2. t_ 従業員のサブタイプ「t_ 営業部員」を削除します。
3. t_ 従業員を削除します。
4. t_A 社従業員のサブタイプ「t_A 社営業部員」を削除します。
5. t_A 社従業員を削除します。

抽象データ型のサブタイプを削除できないケースについては，「(5) 注意」を参照してください。

## (5) 注意

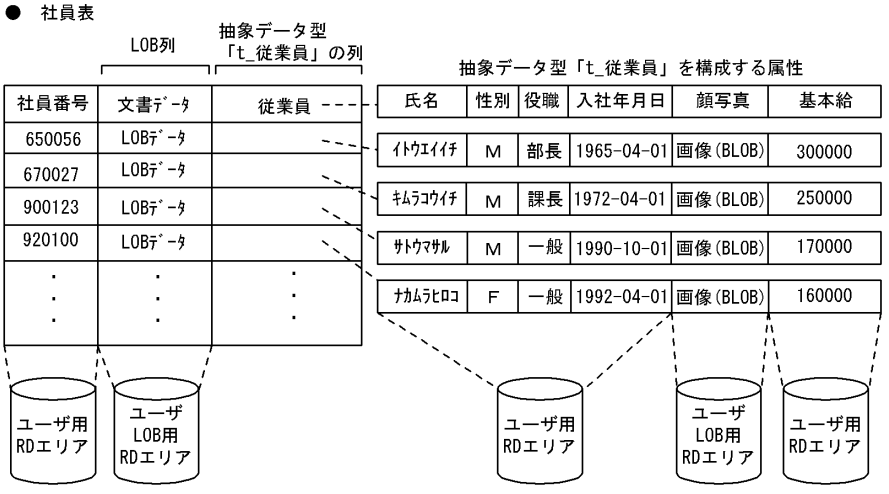
1. コンストラクタ関数を指定して値を生成している場合，抽象データ型を構成する各属性の値がナル値であっても，抽象データ型全体としての値はナル値にはなりません。
2. ある抽象データ型及びそのスーパタイプが表で定義されている場合，その抽象データ型のサブタイプは削除できません。
3. ある抽象データ型及びそのスーパタイプが，ほかの抽象データ型の属性に指定されている場合，その抽象データ型のサブタイプは削除できません。
4. サブタイプを定義する場合，作成するデータ型の上位のデータ型に次に示す条件を満足するストアドプロシジャ及びストアドファンクションがあると，それらは無効状態になります。
   - ストアドプロシジャ及びストアドファンクションの SQL パラメタに指定したデータ型
   - 関数の戻り値のデータ型
   - ストアドプロシジャ及びストアドファンクションから呼び出している関数の引数及び戻り値のデータ型
   - ストアドプロシジャ及びストアドファンクション中で指定しているデータ型（コンポネント指定で抽

象データ型をアクセスする場合，その途中のデータ型を含みます）

## 6.5.2 表の定義

表を構成する列のデータ型の違いによって，RD エリアに格納する単位が異なります。ここでは，社員No，文書データ（LOB データ）及び抽象データ型「t_ 従業員」から構成される社員表の例について説明します。

なお，抽象データ型の列を含む表のうち，抽象データ型の列を除いた部分で構成される表を**抽象データ型列構成基表**といいます。




注
・一つのユーザＬＯＢ用ＲＤエリアには，表中の一つのＬＯＢ列だけを格納します。また，一つの表に複数のＬＯＢ列がある場合には，それぞれ別のユーザＬＯＢ用ＲＤエリアに格納する必要があります。ＬＯＢ列以外の列は複数のユーザ用ＲＤエリアに分割して格納する必要はありません。
・ＬＯＢ列が横分割表の場合には，ＬＯＢ列ごとにユーザＬＯＢ用ＲＤエリアと，表を格納しているユーザ用ＲＤエリアの数を１対１に対応させる必要があります。

〔説明〕

社員表をディスク A,B のユーザ用 RD エリア RDAREA01 と RDAREA02 に，社員表中の文書データ（LOB 列）をユーザ LOB 用 RD エリア LOBAREA01 と LOBAREA02 に，社員表中の抽象データ型（LOB 属性）顔写真の列をユーザ LOB 用 RD エリア LOBAREA03 と LOBAREA04 に分割して格納します。

### (1) キーレンジ分割の場合

**格納条件指定**

```
CREATE TABLE 社員表
(社員番号 CHAR(6),
 文書データ BLOB(64K) IN ((LOBAREA01),(LOBAREA02)),
 従業員 t_従業員 ALLOCATE(顔写真
     IN ((LOBAREA03),(LOBAREA04)))
    )IN ((RDAREA01)社員番号<=700000,(RDAREA02));
```

**境界値指定**

```
CREATE TABLE 社員表
(社員番号 CHAR(6),
 文書データ BLOB(64K) IN ((LOBAREA01),(LOBAREA02)),
```

```
    従業員 t_従業員 ALLOCATE(顔写真
        IN ((LOBAREA03),(LOBAREA04)))
       )PARTITIONED BY 社員番号
        IN ((RDAREA01)800000,(RDAREA02));
```

### (2) フレキシブルハッシュ分割，FIX ハッシュ分割の場合

```
CREATE TABLE 社員表
(社員番号 CHAR(6),
 文書データ BLOB(64K) IN ((LOBAREA01),(LOBAREA02)),
 従業員 t_従業員 ALLOCATE(顔写真
     IN ((LOBAREA03),(LOBAREA04)))
    )[FIX]※ HASH HASH6 BY 社員番号
     IN (RDAREA01,RDAREA02);
```

注※　FIX ハッシュ分割の場合に指定します。

## 6.5.3 インデクスの定義

「社員番号」列にインデクスを定義します。なお，抽象データ型の列にはインデクスを定義できません。

(例)

```
CREATE INDEX INDX1 ON 社員表 (社員番号)
 IN ((RDAREA03),(RDAREA04));
```

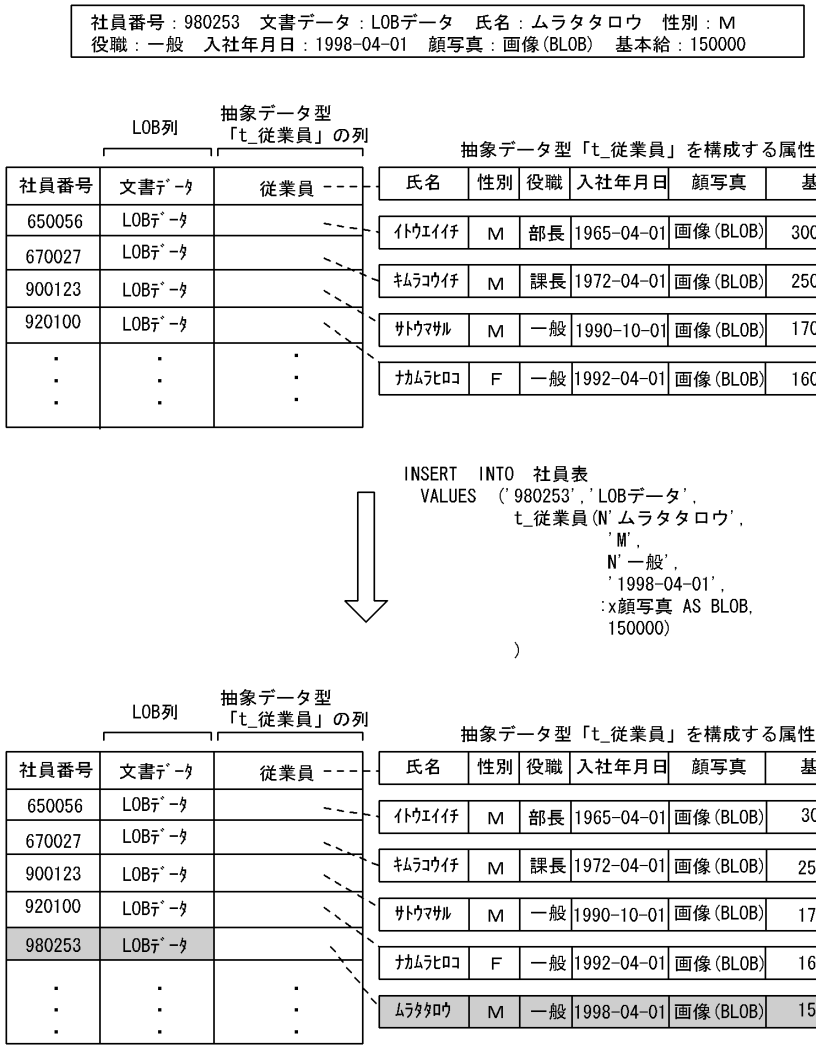
〔説明〕

分割キーインデクス INDX1 を横分割した社員表に対応させて，ユーザ用 RD エリア RDAREA03 と RDAREA04 に分割して格納します。なお，インデクス INDX1 を構成する列に社員番号を指定しています。

## 6.5.4 表へのデータの格納

ユーザが定義した抽象データ型を定義した表にデータを格納するには，操作系 SQL の INSERT 文を実行します。データベース作成ユティリティ（pdload）でデータロードはできません。データを挿入するには，関数を定義して値を生成したものを INSERT 文で挿入します。抽象データ型の列を含む表にデータを挿入するときの手順を次の図に示します。

**図 6-4 抽象データ型の列を含む表にデータを挿入する手順**




注　:x顔写真はBLOB型の埋込み変数で，顔写真の画像が設定されているものとします。

## 6.5.5 データベースの更新ログ取得方式

### (1) データベースの更新ログ取得方式の種類

データベースの更新ログ取得方式には，次に示す 3 種類のモードがあります。

1. **ログ取得モード**
   ロールバック及びロールフォワードに必要なデータベース更新ログを取得します。データ件数が少ない場合の追加データ作成，再編成の場合に使用します。
2. **更新前ログ取得モード**

ロールバックに必要なデータベース更新ログだけを取得します。データ件数が多いときの初期作成，追加データ作成及び再編成の場合に使用します。

3. **ログレスモード**
   データベース更新ログを取得しません。一つの RD エリアに一つの表だけ（分割格納している場合は一つの横分割表）格納している場合で，関連するインデクスも一つの RD エリア内にあるときの初期作成，再編成の場合に使用します。

### (2) データベースの更新ログ取得方式の指定方法

データベースの更新ログ取得方式の指定方法は，次に示すものがあります。

- **クライアント環境定義の** PDDBLOG **に指定する方法**※1
- CREATE TABLE **の** RECOVERY **オペランドに指定する方法**※2

注※1
ユーザ用 RD エリアを更新する UAP の，データベースの更新ログ取得方式を指定するオペランドです。

注※2
ユーザ LOB 用 RD エリアを更新する UAP の，データベースの更新ログ取得方式を指定するオペランドです。

**注意事項**
ユーザ LOB 用 RD エリアのデータベースの更新ログ取得方式（CREATE　TABLE の RECOVERY オペランド指定値）は，クライアント環境定義の指定値によって変わることがあります。クライアント環境定義によって変わるユーザ LOB 用 RD エリアのデータベースの更新ログ取得方式を次の表に示します。

**表 6-6　クライアント環境定義によって変わるユーザ LOB 用 RD エリアのデータベースの更新ログ取得方式**

| クライアント環境定義 PDDBLOG | CREATE TABLE の RECOVERY オペランドの指定値 | | |
|---|---|---|---|
| | ALL | PARTIAL | NO |
| ALL | ALL | PARTIAL | NO |
| NO | NO | NO | NO |

（凡例）
ALL：ログ取得モード
PARTIAL：更新前ログ取得モード
NO：ログレスモード

例えば，CREATE TABLE の RECOVERY オペランドに「PARTIAL」を指定し，クライアント環境定義のログ取得モードが「NO」の場合，NO（ログレスモード）がユーザ LOB 用 RD エリアに設定されます。

## 6.5.6 データの格納状態の確認

抽象データ型の列を含む表にデータを挿入した場合は，運用を開始する前にデータベース状態解析ユティリティ（pddbst）を実行して，データの格納状態を確認することをお勧めします。設計どおりにデータベースを作成できたかどうかを確認できます。

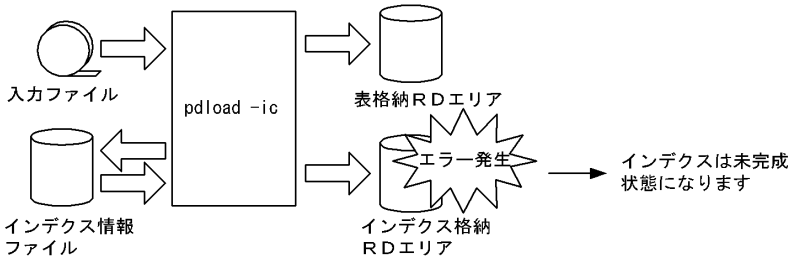
データベース状態解析ユティリティ（pddbst）を実行すると，RD エリア単位のデータの格納状態（物理

解析だけ）の情報を取得できます。

# 6.6 インデクス一括作成中に発生したエラーの対処方法

インデクス一括作成中にエラーが発生した場合，表へのデータ格納は完了しても，インデクスは未完成という不整合な状態になります。ここでは，この状態を回復する方法について説明します。データベース作成ユティリティ（pdload）でインデクス一括作成中にエラーが発生した場合のインデクスの状態を次の図に示します。

**図 6-5 データベース作成ユティリティ（pdload）でインデクス一括作成中にエラーが発生した場合のインデクスの状態**




注 ログレスモードの場合は，表及びインデクス格納RDエリアはログレス閉塞状態になります。ただし，KFPL00703-Iメッセージが出力されている場合は，表へのデータ格納は完了しています。このため，インデクス格納RDエリアと表格納RDエリアが別の場合は，表格納RDエリアの閉塞は解除してかまいません。

## 6.6.1 ログ取得モード又は更新前ログ取得モードでデータロードをしていた場合

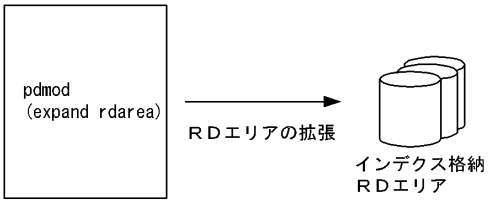
更新前ログ取得モード又はログ取得モードでデータロードをしていた場合の対処方法を説明します。

なお，プラグインインデクスを定義している場合，そのプラグインにプラグインインデクス一括作成部分回復機能があることを前提とします。プラグインにプラグインインデクス一括作成部分回復機能がない場合は，「6.6.2 ログレスモードでデータロードをしていた場合」を参照してください。

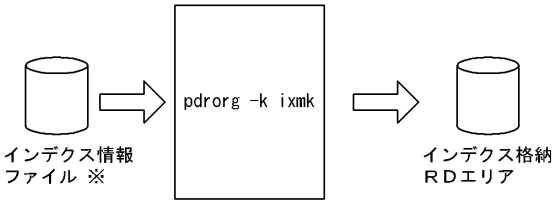
発生したエラー原因に応じて，インデクス格納 RD エリアを回復します。次に示す手順で回復してください。

### (1) インデクス格納 RD エリアの容量不足がエラー原因の場合

1. インデクス格納 RD エリアを拡張します。

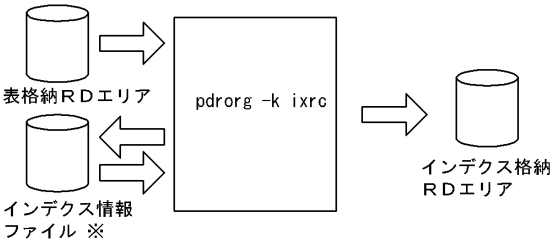



2. インデクスを作成します。データベース作成ユティリティ（pdload）が作成したインデクス情報ファイルを基にインデクスを一括作成します。




注※　このファイルには，追加ロードしたキーと既存データのキー情報が出出力されています。
プラグインインデクスの場合は，追加ロードしたデータのキー情報だけが出力されています。

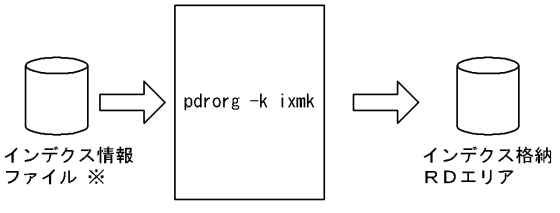
なお，データベース作成ユティリティ（pdload）が作成したインデクス情報ファイルが残っていない場合は，データベース再編成ユティリティ（pdrorg）でインデクスを再作成します。




注※　このファイルは，データベース再編成ユティリティ(pdrorg)が新たに作成します。
追加ロードしたキーと既存データのキー情報を出力します。

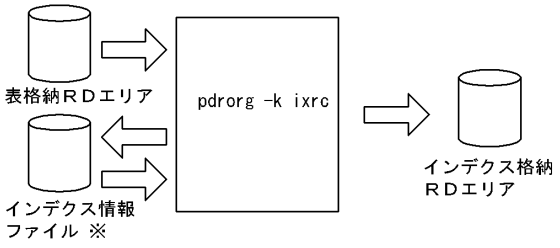
### (2) ソート処理エラー（KFPL15062-E メッセージが出力）又は pdcancel コマンドによるユティリティの強制終了が原因の場合

1. インデクスを作成します。データベース作成ユティリティ（pdload）が作成したインデクス情報ファイルを基にインデクスを一括作成します。




注※　このファイルには，追加ロードしたキーと既存データのキー情報が出出力されています。
プラグインインデクスの場合は，追加ロードしたデータのキー情報だけが出力されています。

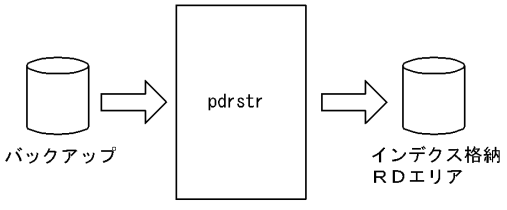
なお，データベース作成ユティリティ（pdload）が作成したインデクス情報ファイルが残っていない場合は，データベース再編成ユティリティ（pdrorg）でインデクスを再作成します。



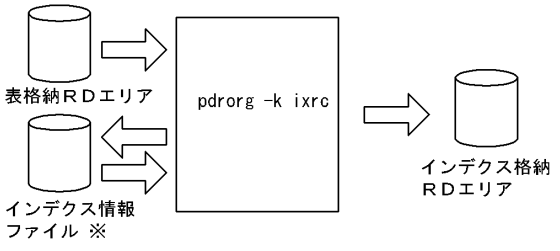

注※　このファイルは，データベース再編成ユティリティ(pdrorg)が新たに作成します。
追加ロードしたキーと既存データのキー情報を出力します。

### (3) ディスク障害が原因の場合

1. 障害となったディスクを交換し，バックアップからユティリティ実行前の状態に戻します。ただし，手順2で実行するインデクスロードが完了するまでアクセスされないように，RD エリアを閉塞状態にしてください。




2. データベース再編成ユティリティ（pdrorg）でインデクスを再作成します。




注※　このファイルは，データベース再編成ユティリティ(pdrorg)が新たに作成します。
追加ロードしたキーと既存データのキー情報を出力します。

## 6.6.2 ログレスモードでデータロードをしていた場合

ログレスモードでデータロードをしていた場合又はプラグインにプラグインインデクス一括作成部分回復機能がない場合の対処方法を説明します。

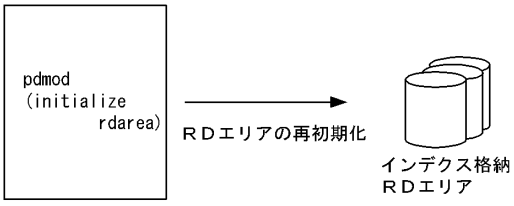
発生したエラー原因に応じて，インデクス格納 RD エリアを回復します。次に示す手順で回復してください。

### (1) インデクス格納 RD エリアの容量不足がエラー原因の場合

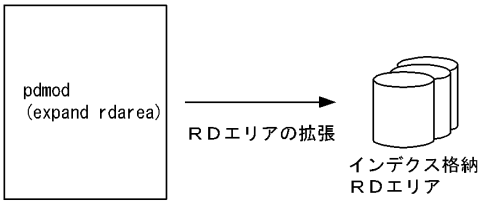
1. インデクス格納 RD エリアを再初期化します。
なお，バックアップからも回復できますが，その場合はインデクスロードが完了するまでアクセスされ

ないよう，RD エリアを閉塞状態にしてください。また，次の場合はバックアップから回復する必要があります。

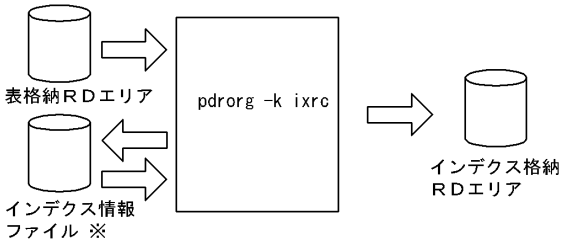
- インデクス格納 RD エリアに該当インデクスとは別の表やインデクス，又は改竄防止表が格納されている




2. インデクス格納 RD エリアを拡張します。



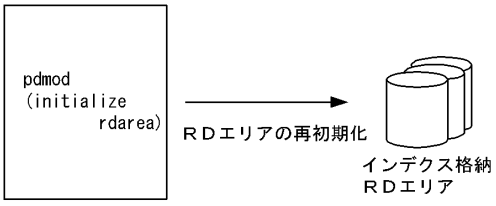

3. データベース再編成ユティリティ（pdrorg）でインデクスを再作成します。



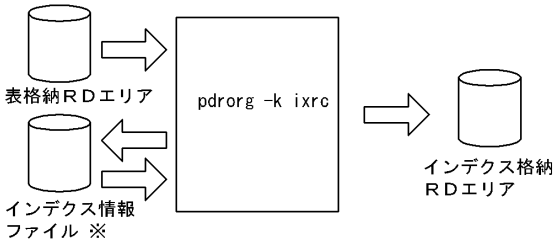

注※　このファイルは，データベース再編成ユティリティ(pdrorg)が新たに作成します。
追加ロードしたキーと既存データのキー情報を出力します。

### (2) ソート処理エラー（KFPL15062-E メッセージが出力）又は pdcancel コマンドによるユティリティの強制終了が原因の場合

1. インデクス格納 RD エリアを再初期化します。
なお，バックアップからも回復できますが，その場合はインデクスロードが完了するまでアクセスされないよう，RD エリアを閉塞状態にしてください。また，次の場合はバックアップから回復する必要があります。
   - インデクス格納 RD エリアに該当インデクスとは別の表やインデクス，又は改竄防止表が格納されている



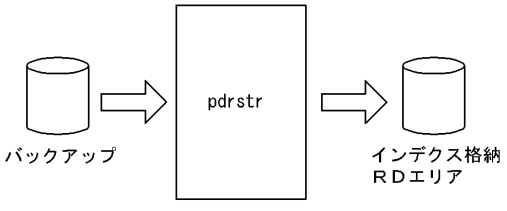

2. データベース再編成ユティリティ（pdrorg）でインデクスを再作成します。



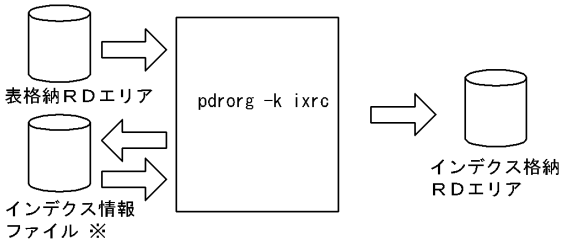

注※　このファイルは，データベース再編成ユティリティ(pdrorg)が新たに作成します。
追加ロードしたキーと既存データのキー情報を出力します。

## (3) ディスク障害が原因の場合

1. 障害となったディスクを交換し，バックアップからユティリティ実行前の状態に戻します。




2. データベース再編成ユティリティ（pdrorg）でインデクスを再作成します。




注※　このファイルは，データベース再編成ユティリティ(pdrorg)が新たに作成します。
追加ロードしたキーと既存データのキー情報を出力します。

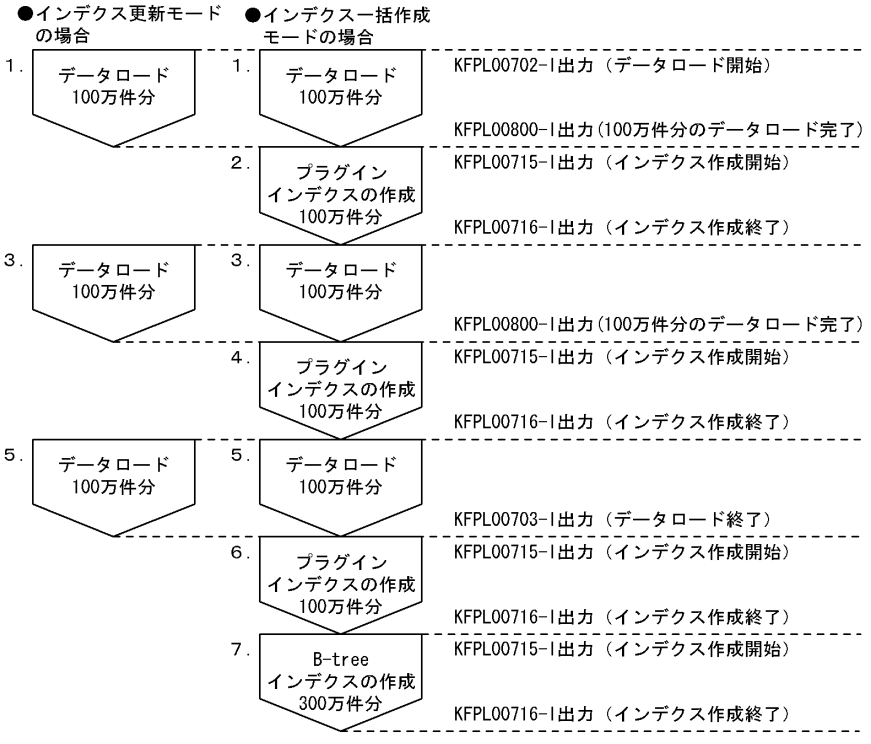
# 6.7 同期点指定のデータロード実行中にユティリティが異常終了したときの対処方法

ここでは，**同期点指定のデータロード**の実行中に，データベース作成ユティリティが異常終了したときの対処方法を説明します。

## 6.7.1 対処方法の概要

異常終了したタイミングによって対処方法が異なります。同期点指定のデータロード実行中にユティリティが異常終了したときの対処方法を次の図に示します。

**図 6-6 同期点指定のデータロード実行中にユティリティが異常終了したときの対処方法**




〔説明〕

- 総データロード件数を 300 万件，同期点行数を 100 万件としています。
- 1，3，5 の時点でユティリティが異常終了した場合は，データロードを再実行してください。
- 2，4 の時点でユティリティが異常終了した場合は，作成に失敗したプラグインインデクスをデータベース再編成ユティリティのインデクス一括作成機能（-k ixmk）で作成してください。その後，データロードを再実行してください。
- 6 の時点でユティリティが異常終了した場合は，作成に失敗したプラグインインデクスをデータベース再編成ユティリティのインデクス一括作成機能（-k ixmk）で作成してください。その後，データベース再編成ユティリティのインデクス再作成機能（-k ixrc）で B-tree インデクスを作成してください。
- 7 の時点でユティリティが異常終了した場合，インデクス情報ファイルが作成されていれば

（KFPL00710-I メッセージが出力されていれば），データベース再編成ユティリティのインデクス一括作成機能（-k ixmk）で B-tree インデクスを作成してください。インデクス情報ファイルが作成されていなければ，データベース再編成ユティリティのインデクス再作成機能（-k ixrc）で B-tree インデクスを作成してください。

## 6.7.2 例題

300 万件のデータロード中にデータベース作成ユティリティが異常終了しました。インデクス一括作成モードで，同期点行数は 100 万件とします。

### (1) メッセージを確認します

次に示すメッセージが出力されています。

```
KFPL00800-I Loading until 2000000th row committed

KFPL00710-I Index information file assigned, index=k87m271."INDX01",
RDAREA="LOB02", file=/pdrorg/INDX01_2

KFPL00715-I Index load started at bes2, index=k87m271."INDX01", RDAREA="LOB02"
```

〔説明〕

- KFPL00800-I メッセージから，200 万件までデータロード処理が完了していることが分かります。
- KFPL00715-I メッセージから，プラグインインデクスの作成処理が開始されていることが分かります。それに対応する完了メッセージ（KFPL00716-I）が出力されていません。

これによって，100 ～ 200 万件のプラグインインデクス作成中に，ユティリティが異常終了したことが分かります。

### (2) pdrorg コマンドでプラグインインデクスを一括作成します

データベース再編成ユティリティで，100 万件（100 ～ 200 万件）分のプラグインインデクスを一括作成します。

```
pdrorg -k ixmk -t TABLE1 C:¥pdrorg¥rorg01
```

〔説明〕

-k：プラグインインデクスを一括作成するため ixmk を指定します。

-t：表の名称を指定します。

C:¥pdrorg¥rorg01：

pdrorg コマンドの制御文ファイル名を指定します。制御文ファイルの内容を次に示します。制御文ファイル中に指定するインデクス情報ファイルは (1) で出力される KFPL00710-I メッセージから分かります。

```
index INDX01 LOB02 C:¥pdrorg¥INDX01_2
```

### (3) データロードを再実行します

```
pdload TABLE1 C:¥pdload¥load01
```

〔説明〕

オプションの指定を変更する必要はありません。

### (4) データロード対象 RD エリアのバックアップを取得します

データロード対象 RD エリアのバックアップを取得します。RD エリア単位のバックアップの取得方法については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

### (5) pdrels コマンドでデータロード対象 RD エリアの閉塞を解除します

```
pdrels -r DATA01,DATA02,DATA03,INX01,INX02,INX03,LOB01,LOB02,LOB03
```

コマンドの実行後，実行結果が正しいかどうか確認することをお勧めします。コマンドの実行結果の確認方法については，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

# 7 ほかの製品との連携

この章では，HiRDB とほかの製品とを連携する方法について説明します。


7.1 **レプリケーション機能との連携**

7.2 OLTP **との連携**

7.3 HiRDB External Data Access **機能との連携**

7.4 JP1 **との連携**

# 7.1 レプリケーション機能との連携

HiRDB のレプリケーション機能（HiRDB Datareplicator，HiRDB Dataextractor）を使用するときに指定する項目について説明します。

## 7.1.1 HiRDB Datareplicator との連携

HiRDB Datareplicator を使用すると，HiRDB のデータベースの更新に連動して自動的にデータを抽出し，そのデータベース更新内容をほかの HiRDB のデータベースに反映できます。HiRDB Datareplicator を使用するには，HiRDB システム定義で次に示すオペランドを指定します。

- pd_rpl_init_start オペランド
  HiRDB Datareplicator 連携機能を HiRDB 開始時から使用するかどうかを指定します。
- pd_rpl_hdepath オペランド
  抽出側 HiRDB Datareplicator 運用ディレクトリ名を指定します。ここで指定するディレクトリ名は，抽出側 HiRDB Datareplicator の環境変数 HDEPATH で指定した名称にしてください。
- pd_log_rpl_no_standby_file_opr オペランド
  HiRDB Datareplicator 連携機能の使用時に，HiRDB Datareplicator でのシステムログの抽出が完了していないため，すべてのシステムログファイルがスワップ先にできない状態でスワップ要求が発生した場合の運用方法を指定します。

HiRDB Datareplicator を使用してデータ連動する場合のシステムの環境設定及び運用方法については，マニュアル「HiRDB データ連動機能 HiRDB Datareplicator Version 8」を参照してください。

**注意事項**

- HiRDB Datareplicator がサポートしていない HiRDB の機能を使用した場合，データ連動機能が使えなくなることがあります。詳細については，マニュアル「HiRDB データ連動機能 HiRDB Datareplicator Version 8」を参照してください。また，最新の情報については，次のサイトで公開しているオンラインマニュアルを参照してください。
  http://www.hitachi.co.jp/Prod/comp/soft1/manual/common/hirdb.htm
- 回復不要 FES を使用する場合
  回復不要 FES では，反映側 HiRDB Datareplicator の同期点処理方式に二相コミット方式を利用（反映システム定義 commitment_method オペランドに fxa_sqle を指定）した反映処理を実行できないため，回復不要 FES 以外のフロントエンドサーバを使用して反映処理を実行する必要があります。詳細は，「9.1.4 回復不要 FES」を参照してください。

## 7.1.2 HiRDB Dataextractor との連携

HiRDB Dataextractor を使用すると，メインフレーム又は HiRDB のデータベースのデータをまとめて抽出し，HiRDB のデータベースへ順次格納できます。HiRDB Dataextractor の特長を次に示します。

- 基幹データベースのある時点の情報を部門データベースに一括して反映できます。これによって，部門データベースの表の初期作成又は全データのリフレッシュができます。
- 基幹データベースから部分的にデータを抽出し，各業務に適した部門データベースを作成できます。

HiRDB Dataextractor については，マニュアル「データベース抽出・反映サービス機能 HiRDB Dataextractor Version 8」を参照してください。

# 7.2 OLTP との連携

ここでは，HiRDB が X/Open XA インタフェースを使用して，OLTP と連携する方法について説明します。ここで説明する項目は次のとおりです。

1. OLTP と連携できる製品
2. HiRDB XA ライブラリ
3. OLTP と連携した HiRDB システムの構成例
4. トランザクションの移行
5. トランザクションマネジャへの登録
6. トランザクションマネジャに登録する情報
7. トランザクションマネジャへの登録例
8. トランザクションマネジャへの登録の変更
9. トランザクションマネジャと HiRDB 間のコネクションが切断されたときの再接続方法
10. 注意事項

## 7.2.1 OLTP と連携できる製品

HiRDB では次に示す OLTP 製品と連携できます。

- OpenTP1
- TPBroker for C++
- TUXEDO
- WebLogic Server

ただし，適用 OS が Windows (x64) の場合，64 ビットモードで動作する OLTP 製品用のクライアントライブラリを用意していないので連携できません。

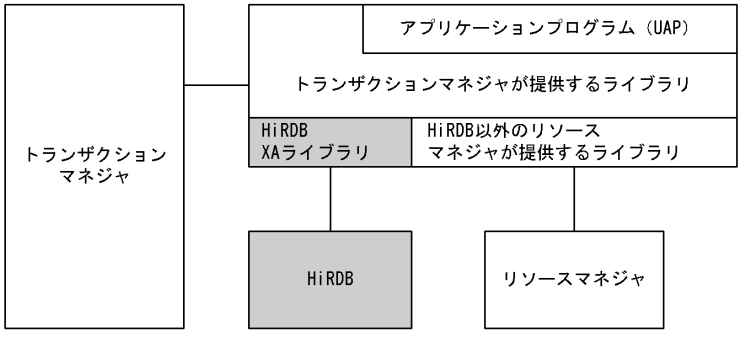
## 7.2.2 HiRDB XA ライブラリ

X/Open XA **インタフェース**とは，分散トランザクション処理（DTP：Distributed Transaction Processing）システムのトランザクションマネジャ（TM：Transaction Manager）とリソースマネジャ（RM：Resource Manager）の接続インタフェースを規定した X/Open の標準仕様です。XA インタフェースを使用すると，リソースマネジャのトランザクション処理をトランザクションマネジャで制御できます。リソースマネジャのトランザクション処理をトランザクションマネジャで制御するには，リソースマネジャが提供するライブラリとトランザクションマネジャが提供するライブラリを UAP にリンケージします。

HiRDB の UAP の処理をトランザクションマネジャで制御するために，HiRDB は HiRDB XA **ライブラリ**を提供しています。HiRDB XA ライブラリは，X/Open DTP ソフトウェアアーキテクチャの XA インタフェースの仕様に準拠しています。

X/Open DTP モデルでの HiRDB の位置づけを次の図に示します。

**図 7-1 X/Open DTP モデルでの HiRDB の位置づけ**




なお，X/Open XA インタフェースを使用する UAP から回復不要 FES に接続すると，SQL がエラーリターンします。クライアント環境定義の PDFESHOST 及び PDSERVICEGRP を指定して，回復不要 FES ではないフロントエンドサーバに接続してください。

## （1）HiRDB XA ライブラリが提供する機能

HiRDB XA ライブラリが提供する機能を次の表に示します。

**表 7-1 HiRDB XA ライブラリが提供する機能**

| 機能 | 説明 |
|---|---|
| トランザクションの移行 | トランザクションのコミット処理を UAP が HiRDB にアクセスしたときと異なるプロセスで実行する機能です。ここでいう UAP とは，HiRDB XA ライブラリを使用して HiRDB に接続する UAP のことです。<br>トランザクションの移行を使用するかどうかは，クライアント環境定義の PDXAMODE オペランドで指定します。トランザクションの移行については，「7.2.4　トランザクションの移行」を参照してください。 |
| 一相最適化 | 二相コミット制御を一相に最適化する機能です。<br>一相最適化を使用する場合，トランザクションの完了種別がトランザクションマネジャと HiRDB で一致しないことがあります。詳細については，「7.2.10(3) 一相最適化に関する注意事項」を参照してください。 |
| 読み取り専用 | プリペア要求で HiRDB のリソースが更新されていない場合，トランザクションマネジャが二相目にコミット要求をしないで最適化する機能です。 |
| 動的トランザクションの登録 | UAP を実行する直前に，HiRDB が動的にトランザクションを登録する機能です。 |
| 複数接続機能 | 一つのプロセスから HiRDB サーバに対して複数の CONNECT を別々に実行する機能です。X/Open XA インタフェース環境下での複数接続機能については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。 |

注

HiRDB XA ライブラリでは，非同期 XA 呼び出し（トランザクションマネジャが非同期に HiRDB XA ライブラリを呼び出す機能）を提供していません。

## （2）マルチスレッド対応の XA インタフェース

マルチスレッド対応の XA インタフェースを利用すると，TPBroker for C++ と HiRDB で OTS（Object Transaction Service）連携ができます。

なお，マルチスレッド用のライブラリは C 言語又は C++ 言語でだけ使用できます。COBOL 言語では使

用できません。

マルチスレッド対応の XA インタフェースを使用するには専用の HiRDB クライアントライブラリをリンクしてください。バージョン 05-06 より前の HiRDB クライアントライブラリはマルチスレッドに対応していません。マルチスレッド用のライブラリは，既存の HiRDB クライアントが接続できるすべての HiRDB サーバに対して使用できます。マルチスレッド用のライブラリを使用すると，スレッドの間で接続を共有できます。

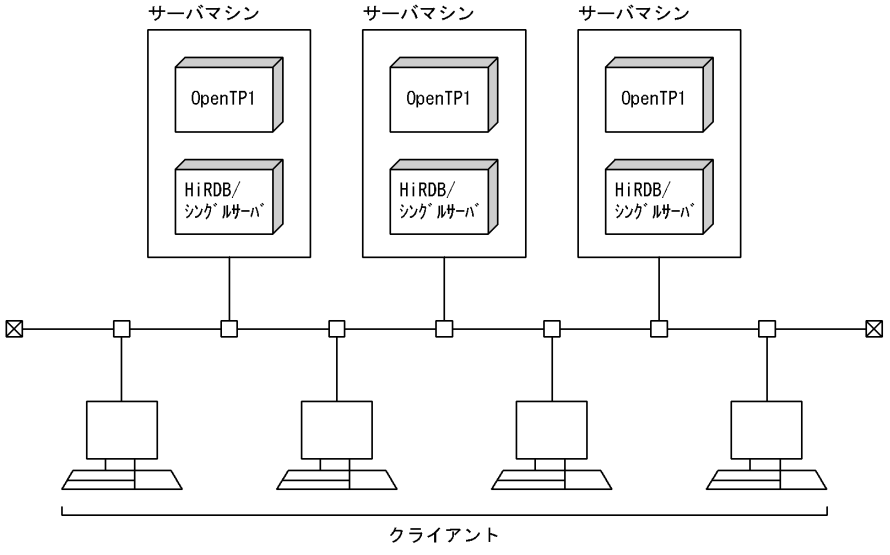
## 7.2.3　OLTP と連携した HiRDB システムの構成例

OLTP と連携した HiRDB システムについて，OpenTP1 を使用した例で説明します。

### （1）HiRDB/ シングルサーバとの連携

HiRDB/ シングルサーバと OLTP（OpenTP1）を連携すると，複数の HiRDB/ シングルサーバの更新処理を一つのトランザクションとして実行できます。この場合，データベースはキーレンジ分割などで分割配置し，各サーバマシンで稼働する OLTP（OpenTP1）が，各 HiRDB/ シングルサーバへ処理を振り分けます。処理を振り分けることで，高速にトランザクション処理を実行できるようにしています。複数の HiRDB/ シングルサーバを統合化する場合は，OLTP との連携を検討してください。HiRDB/ シングルサーバと OLTP（OpenTP1）との連携を次の図に示します。

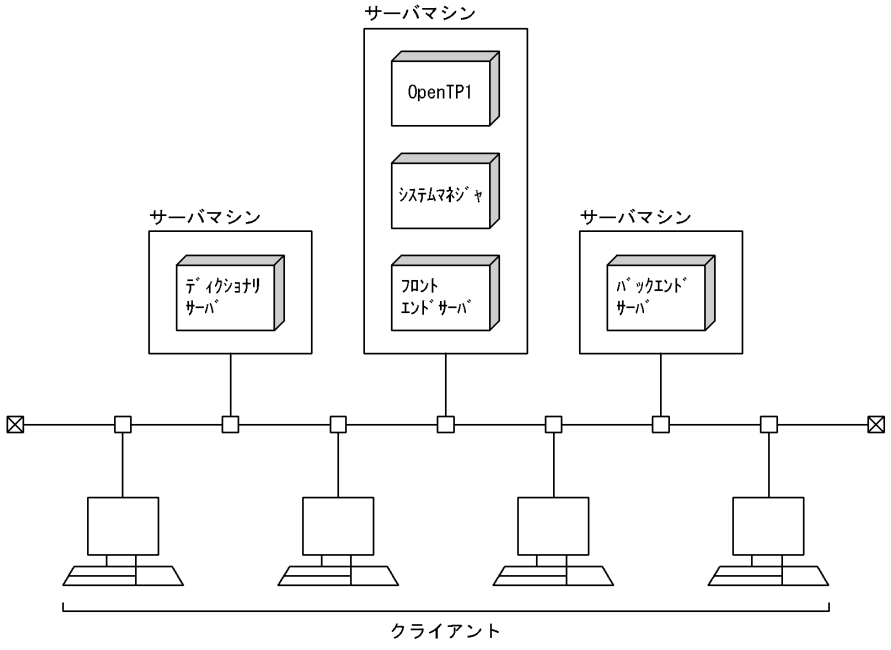
**図 7-2　HiRDB/ シングルサーバと OLTP（OpenTP1）との連携**




### （2）HiRDB/ パラレルサーバとの連携

HiRDB/ パラレルサーバと OLTP（OpenTP1）を連携すると，HiRDB/ パラレルサーバで負荷分散したデータベースの更新処理をトランザクション処理として実行できます。HiRDB/ パラレルサーバと OLTP（OpenTP1）との連携を次の図に示します。

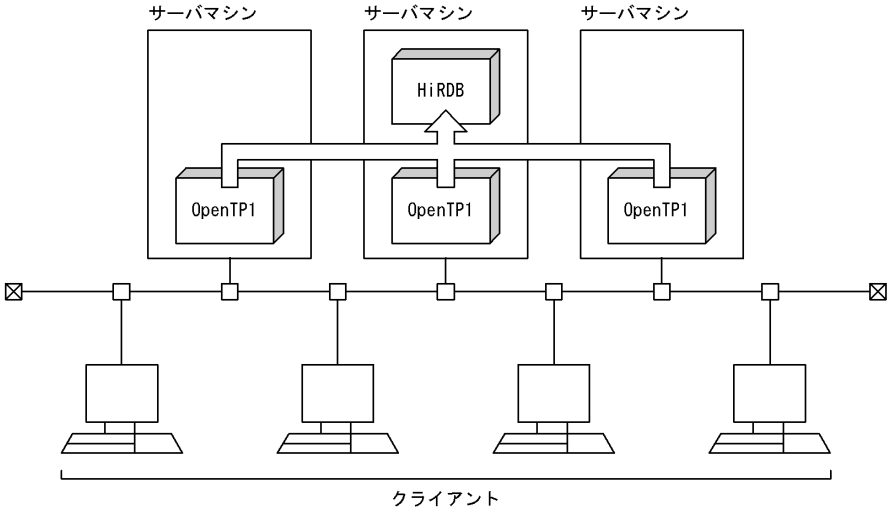
**図 7-3 HiRDB/パラレルサーバと OLTP（OpenTP1）との連携**




## (3) 複数の OLTP（OpenTP1）と HiRDB との連携

複数の OLTP（OpenTP1）と一つの HiRDB 間でクライアント／サーバ型で通信して連携する形態です。異なる OLTP（OpenTP1）から一つの HiRDB に，同時に接続できます。この場合，各 OLTP（OpenTP1）の OLTP 識別子（クライアント環境定義の PDTMID）が異なるようにしてください。複数の OLTP（OpenTP1）と HiRDB との連携を次の図に示します。

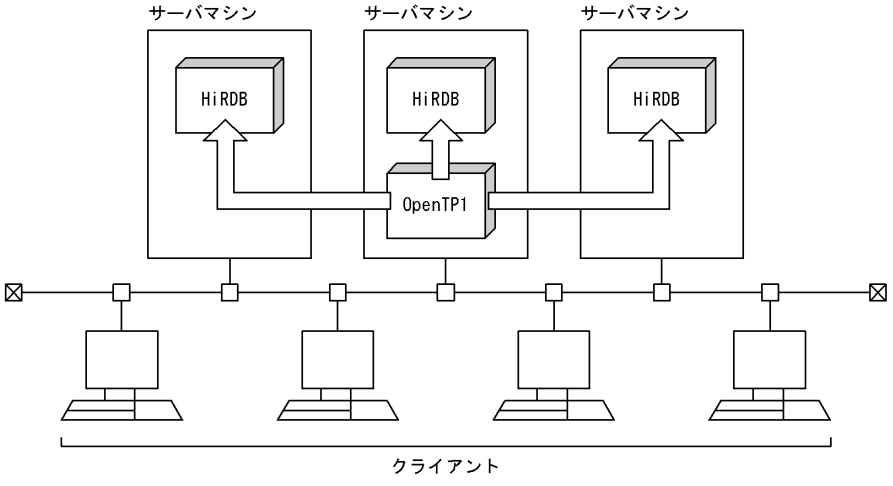
**図 7-4 複数の OLTP（OpenTP1）と HiRDB との連携**



### （4）一つの OLTP（OpenTP1）と複数の HiRDB との連携

一つの OLTP（OpenTP1）と複数の HiRDB で連携する形態です。異なるサーバマシンの HiRDB に同時に接続して SQL 文を実行できます。この場合，複数接続機能を使用する必要があります。一つの OLTP（OpenTP1）と複数の HiRDB との連携を次の図に示します。

**図 7-5　一つの OLTP（OpenTP1）と複数の HiRDB との連携**

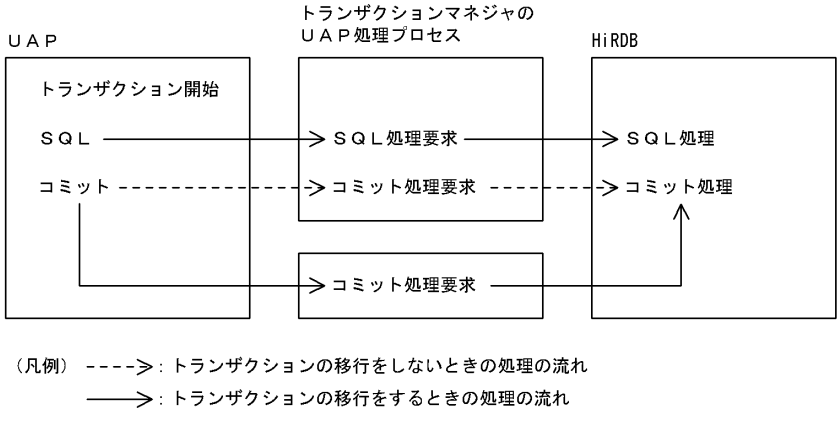



複数接続機能については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

## 7.2.4　トランザクションの移行

HiRDB XA ライブラリを使用して HiRDB に接続する UAP では，HiRDB にアクセスしたときと異なるプロセスでトランザクションのコミット処理を実行できます。このことを**トランザクションの移行**といいます。トランザクションの移行の概要を次の図に示します。

**図 7-6　トランザクションの移行の概要**




**メリット**

トランザクションの移行を使用すると，トランザクションマネジャの UAP 処理はトランザクション

の完了を待たなくても次回のサービス要求を受け付けられます。このため，この機能を使用しないときに比べて，少ないプロセス数で UAP を実行できます。ただし，HiRDB が使用するサーバプロセス数及び排他待ち※回数が増える場合があります。

注※

トランザクションが完了するまでの間，次回のサービス要求による HiRDB へのアクセスは，排他待ちになる場合が増えます。

**適用基準**

トランザクションマネジャがトランザクションの移行を使用する場合は，HiRDB もトランザクションの移行を使用してください。

トランザクションマネジャがトランザクションの移行を使用しない場合は，HiRDB もトランザクションの移行を使用しないでください。

トランザクションマネジャがトランザクションの移行の使用可否を設定できる場合は，次に示す点に考慮してトランザクションの移行の使用可否を設定してください。

- **HiRDB へのアクセス負荷より，UAP 自身の処理の負荷が大きい場合にトランザクションの移行を使用します。**

**運用方法**

トランザクションの移行を使用する場合は，クライアント環境定義の PDXAMODE オペランドに 1 を指定してください。この機能を使用しない場合は，このオペランドに 0 を指定するか，このオペランドを省略してください。

PDXAMODE オペランドについては，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

**注意事項**

1. トランザクションマネジャと HiRDB の間で，トランザクションの移行を使用するかどうかの設定が合っていないと，トランザクションが決着できなかったり，HiRDB が異常終了したり，トランザクションマネジャにエラーリターンしたりすることがあります。
2. この機能を使用すると，LOCK 文の LOCK TABLE UNTIL DISCONNECT の有効範囲が変わります。LOCK TABLE UNTIL DISCONNECT の有効範囲については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

## (1) トランザクションマネジャが OpenTP1 の場合

トランザクションの移行を使用すると，OpenTP1 のコミット最適化及びプリペア最適化に HiRDB が対応します。したがって，OpenTP1 の trnstring オペランドの -d オプションを省略した場合は，この機能を使用してください。-d オプションを指定した場合は，この機能を使用しないでください。

OpenTP1 システム定義のトランザクションサービス定義の trnstring オペランドと HiRDB の PDXAMODE オペランドの関係を次の表に示します。

**表 7-2 OpenTP1 の trnstring オペランドと HiRDB の PDXAMODE オペランドの関係**

| trnstring オペランドの指定 | PDXAMODE オペランドの値 |
|---|---|
| -d オプションを省略 | 1 |
| -d オプションを指定 | 0 |

（凡例）－：該当しません。

注

- trnstring オペランドと PDXAMODE オペランドの指定が合っていないと，HiRDB がトランザクションを決着できません。このとき，HiRDB は OpenTP1 に対して XA 関数エラーリターンコード（－ 6）を返します。

trnstring オペランドについては，マニュアル「OpenTP1 システム定義」を参照してください。コミット最適化又はプリペア最適化については，マニュアル「OpenTP1 プログラム作成の手引」を参照してください。

## 7.2.5 トランザクションマネジャへの登録

OLTP と連携するには，HiRDB をトランザクションマネジャに登録する必要があります。HiRDB をトランザクションマネジャに登録するには，各トランザクションマネジャのコマンド又は機能を使用します。

- OpenTP1 の場合：trnlnkrm コマンドで HiRDB を登録します。
- TPBroker for C++ の場合：tslnkrm コマンドで HiRDB を登録します。
- TUXEDO の場合：%TUXDIR%¥udataobj¥RM に HiRDB を登録します。%TUXDIR% は，TUXEDO システム・ソフトウェアがあるディレクトリの絶対パス名を示しています。
- WebLogic Server の場合：WebLogic Server の JDBC 接続プールの「ドライバクラス名」と「プロパティ」で HiRDB を登録します。

### (1) 動的登録と静的登録

HiRDB をトランザクションマネジャに登録するときに，次に示すどちらかの方法を選択してください。

- 動的登録
- 静的登録

なお，一つのトランザクションマネジャに対して，動的登録と静的登録を混在して使用できません。

WebLogic Server の場合は静的登録しかできません。

#### (a) 動的登録とは

HiRDB をトランザクションマネジャに動的登録すると，トランザクション内で最初の SQL 文を発行したときに，UAP がトランザクションマネジャの制御下に入ります。UAP が HiRDB を含む複数のリソースマネジャをアクセスする場合，又は UAP が HiRDB をアクセスするとは限らない場合などに，トランザクションマネジャからの HiRDB に対するトランザクション制御のオーバヘッドを削減できます。

#### (b) 静的登録とは

HiRDB をトランザクションマネジャに静的登録すると，UAP が SQL 文を発行するかどうかに関係なく，トランザクションの開始時に常にトランザクションマネジャの制御下に入ります。

トランザクションマネジャが OpenTP1 の場合，UAP と HiRDB とのコネクションが切断されたとき（ユニットの異常終了，又はサーバプロセスの異常終了などのとき）に，OpenTP1 にはトランザクション開始時に再接続をする機能があるため，UAP の再起動が不要になります。

### (2) 動的登録と静的登録の違い

動的登録と静的登録の違いを次の表に示します。

表 7-3 動的登録と静的登録の違い

| 差異のポイント | 動的登録 | 静的登録 |
|---|---|---|
| トランザクション開始時 | 管理しない | • コネクション確立中かの確認<br>• トランザクションマネジャ制御下でのトランザクションの管理を開始<br>• コネクション確立※1 |
| トランザクション内で最初のSQL発行時 | • トランザクションマネジャの制御下での管理を開始<br>• HiRDBのトランザクション開始<br>• SQL処理<br>• コネクション確立※1 | • HiRDBのトランザクション開始<br>• SQL処理 |
| トランザクション処理中のトランザクションマネジャとHiRDB間の通信回数 | SQL文数＋コミット処理通信回数＋1（コネクション確立用通信分）※1 | SQL文数＋コミット処理通信回数＋1（トランザクション開始処理用の通信分）＋1（コネクション確立用通信分）※1 |
| トランザクションマネジャとHiRDB間のコネクションが，途中で切断したときの再接続方法※2 | 次回のトランザクション開始時に自動的に再接続※3 | 次回のトランザクション開始時に自動的に再接続※4 |

注※1

マルチスレッド対応のXAインタフェース使用時にする処理です。

注※2

ネットワーク障害による切断は検知できません。ただし，トランザクションマネジャがTPBroker for C++，又はWeblogic Serverの場合はトランザクション開始時に接続を確立するため，再接続できます。

注※3

トランザクションマネジャがOpenTP1/Server Baseの場合は，OpenTP1/Server Baseのtrn_rm_open_close_scopeオペランドにtransactionを指定することで，ネットワーク障害による切断の場合でも再接続できます。

注※4

トランザクションマネジャがOpenTP1/Server Baseの場合は，ネットワーク障害による切断の場合でも再接続できます。

## 7.2.6 トランザクションマネジャに登録する情報

HiRDBをリソースマネジャとしてトランザクションマネジャに登録する方法については，トランザクションマネジャのマニュアルを参照してください。このとき，次に示す情報をトランザクションマネジャに指定します。

なお，WebLogic Serverの場合は(6)以降の操作を実施してください。

### (1) RMスイッチ名

動的登録にするか又は静的登録にするかは，RMスイッチ名の指定で決まります。HiRDBのRMスイッチ名（xa_switch_t構造体名）を次に示します。

- 動的登録の場合：pdtxa_switch
- 静的登録の場合：pdtxa_switch_y

### (2) RM 名

RM スイッチ（xa_switch_t 構造体）で定義されている RM 名（リソースマネジャ名）は，HiRDB_DB_SERVER です。

### (3) オープン文字列

トランザクションマネジャが xa_open でリソースマネジャをオープンするときに使用するオープン文字列は，**複数接続機能**を使用する場合に指定してください。複数接続機能を使用しない場合はオープン文字列を指定する必要はありません。トランザクションマネジャが TUXEDO 又は WebLogic Server の場合は，複数接続機能を使用できません。

複数接続機能を使用する場合は，トランザクションマネジャに複数の HiRDB を登録し，各 HiRDB に対してオープン文字列を指定します。オープン文字列には次に示す項目を指定します。

- 接続先で有効にする環境変数の設定したファイルの絶対パス名，又は接続先で有効にする環境変数をレジストリ登録したときに指定した環境変数グループ名（環境変数をレジストリ登録する方法については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照）
- 環境変数グループ識別子

次のどちらかの書式で記述します。

- "**環境変数グループ識別子＋環境変数設定ファイル名**又は**環境変数グループ名**"
- "**環境変数グループ識別子＊環境変数設定ファイル名**又は**環境変数グループ名**"

これ以外の形式で指定した場合は，オープン文字列が無視されます。また，環境変数グループ識別子は 4 バイト固定，オープン文字列は全体で 257 バイト以上にできません。

トランザクションマネジャが OpenTP1 又は TPBroker for C++ の場合のオープン文字列の登録例を次に示します。

#### (a) OpenTP1 の場合

OpenTP1 のトランザクションサービス定義の trnstring オペランドでオープン文字列を登録します。ここでは二つの HiRDB を OpenTP1 に登録します。登録条件は次のとおりとします。

| リソースマネジャ | 環境変数グループ識別子 | 環境変数設定ファイル名又は環境変数グループ名 |
|---|---|---|
| HiRDB1 | HDB1 | hirdb11<br>hirdb12 |
| HiRDB2 | HDB2 | hirdb21<br>hirdb22 |

オープン文字列の登録例を次に示します。

```
trnstring -n HiRDB_DB_SERVER -i H1 -o "HDB1*hirdb11" -O "HDB1+hirdb12"
trnstring -n HiRDB_DB_SERVER -i H2 -o "HDB2*hirdb21" -O "HDB2+hirdb22"
```

〔説明〕

-n：リソースマネジャ名を指定します。

-i：リソースマネジャ拡張子を指定します。

-o：トランザクションサービス用 xa_open 関数用文字列を指定します。

OpenTP1 のトランザクションサービスプロセスが使用するオープン文字列を指定します。
"**環境変数グループ識別子＊環境変数設定ファイル名**又は**環境変数グループ名**" の形式で指定します。

-O：ユーザサーバ用 xa_open 関数用文字列を指定します。
ユーザサーバプロセスが使用するオープン文字列を指定します。
"**環境変数グループ識別子＋環境変数設定ファイル名**又は**環境変数グループ名**" の形式で指定します。

- -o と -O には同じ環境変数グループ識別子を指定してください。
- -o と -O に指定するファイル又はレジストリで設定する環境変数は同じ内容にしてください。

**備考**

OpenTP1 のユーザサービス定義の trnrmid オペランドで，ユーザサービスから接続する HiRDB を選択します。HiRDB1 と HiRDB2 に接続する例を次に示します。

```
trnrmid -n HiRDB_DB_SERVER -i H1,H2
```

### (b) TPBroker for C++ の場合

TPBroker for C++ のリソースマネジャ定義の xa_open_string_info オペランドでオープン文字列を登録します。ここでは二つの HiRDB を TPBroker for C++ に登録します。登録条件は次のとおりとします。

| リソースマネジャ | 環境変数グループ識別子 | 環境変数設定ファイル名又は環境変数グループ名 |
|---|---|---|
| HiRDB1 | HDB1 | hirdb11<br>hirdb12 |
| HiRDB2 | HDB2 | hirdb21<br>hirdb22 |

オープン文字列の登録例を次に示します。

```
tsdefvalue /OTS/RM/HiRDB_DB_SERVER_1/DMN/xa_open_string_info     1
-s "HDB1*hirdb11"
tsdefvalue /OTS/RM/HiRDB_DB_SERVER_1/xa_open_string_info         2
-s "HDB1+hirdb12"

tsdefvalue /OTS/RM/HiRDB_DB_SERVER_2/DMN/xa_open_string_info     1
-s "HDB2*hirdb21"
tsdefvalue /OTS/RM/HiRDB_DB_SERVER_2/xa_open_string_info         2
-s "HDB2+hirdb22"
```

〔説明〕

1. /OTS/RM/RM 名 /DMN/xa_open_string_info には，TPBroker for C++ の回復プロセスが使用するオープン文字列を指定します。環境変数グループ識別子と環境変数設定ファイル又は環境変数グループ名の間の文字には＊を指定してください。
2. /OTS/RM/RM 名 /xa_open_string_info には，アプリケーションプログラムプロセス及び決着プロセスが使用するオープン文字列を指定します。環境変数グループ識別子と環境変数設定ファイル又は環境変数グループ名の間の文字には＋を指定してください。

- RM 名が同じ場合は，同じ環境変数グループ識別子を指定してください。
- RM 名が同じ場合は，各環境変数設定ファイル又は環境変数グループに設定する環境変数を同じ内

容にしてください。

- 決着プロセスに対して環境変数 TPRMINFO を設定している場合，/OTS/RM/RM 名 /（TPRMINFO 設定値）/xa_open_string_info に指定するオープン文字列には /OTS/RM/RM 名 / xa_open_string_info と同じ文字列を指定してください。また，決着プロセスに対して TPRMINFO を設定しない場合でも，複数接続機能を使用するときは /OTS/completion_process_env にデフォルトとして 'TPRMINFO=' を指定してください。指定例を次に示します。

  **（例）**tsdefvalue /OTS completion_process_env -a 'TPRMINFO='

## （4）クローズ文字列

トランザクションマネジャが xa_close でリソースマネジャをクローズするときに使用するクローズ文字列は指定不要です。

## （5）RM 関連オブジェクト名

RM 関連オブジェクト名には，次の表に示すライブラリ名を指定します。

**表 7-4　RM 関連オブジェクト名に指定するライブラリ名**

| ライブラリの種類 | ライブラリ名 |
|---|---|
| シングルスレッド用 | pdcltx32.lib |
| シングルスレッド用（複数接続機能対応） | pdcltxs.lib ※ |
| マルチスレッド用（複数接続機能対応） | pdcltxm.lib |

注※

TUXEDO と連携する場合に指定するライブラリ名です。

## （6）クライアント環境定義

トランザクションマネジャに HiRDB のトランザクション処理を制御させるためには，HiRDB のクライアント環境定義をトランザクションマネジャの定義に設定する必要があります。OLTP 環境下でのクライアント環境定義の設定方法については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

### （a）OpenTP1 の場合

トランザクションマネジャが OpenTP1 の場合，クライアント環境定義を次に示す OpenTP1 のシステム定義に指定する必要があります。

- システム環境定義
- ユーザサービスデフォルト定義
- ユーザサービス定義
- トランザクションサービス定義

これらの定義については，マニュアル「OpenTP1 システム定義」を参照してください。

なお，複数の OpenTP1 と接続する場合は，次に示すクライアント環境定義を必ず指定してください。

- HiRDB_PDTMID 又は PDTMID

### （b）TPBroker for C++ の場合

クライアント環境定義は TPBroker for C++ のシステム定義に指定してください。

#### (c) TUXEDO の場合

TUXEDO コンフィギュレーション・ファイル（UBBCONFIG ファイル）の ENVFILE パラメタで指定したファイルに，クライアント環境定義を指定してください。TUXEDO コンフィギュレーション・ファイルについては，TUXEDO のマニュアルを参照してください。

#### (d) WebLogic Server の場合

WebLogic Server プロセスの環境変数にクライアント環境定義を指定してください。

### (7) JDBC ドライバ（WebLogic Server 使用時限定）

HiRDB を登録するときに指定する JDBC ドライバのパッケージ名称と Driver クラス名称を次に示します。

- パッケージ名称：JP.co.Hitachi.soft.HiRDB.JDBC
- Driver クラス名称：JdbhXADataSource

## 7.2.7 トランザクションマネジャへの登録例

### (1) OpenTP1 の場合

HiRDB を OpenTP1 に登録するには，OpenTP1 の trnlnkrm コマンドを使用します。trnlnkrm コマンドの指定例を次に示します。

#### (a) 動的登録の場合

```
trnlnkrm -a HiRDB_DB_SERVER -s pdtxa_switch
-o C:¥win32app¥hitachi¥hirdb_s¥client¥lib¥pdcltx32.lib
```

〔説明〕

-a：RM 名を指定します。

-s：RM スイッチ名（XA スイッチ構造体の名称）を指定します。RM スイッチ名は，登録方法（動的登録又は静的登録）によって異なります。

-o：RM 関連オブジェクト名（共用ライブラリのファイル名）を指定します。

#### (b) 静的登録の場合

```
trnlnkrm -a HiRDB_DB_SERVER -s pdtxa_switch_y
-o C:¥win32app¥hitachi¥hirdb_s¥client¥lib¥pdcltx32.lib
```

〔説明〕

-a：RM 名を指定します。

-s：RM スイッチ名（XA スイッチ構造体の名称）を指定します。RM スイッチ名は，登録方法（動的登録又は静的登録）によって異なります。

-o：RM 関連オブジェクト名（共用ライブラリのファイル名）を指定します。

### (2) TPBroker for C++ の場合

HiRDB を TPBroker for C++ に登録するには，TPBroker for C++ の tslnkrm コマンドを使用します。tslnkrm コマンドの指定例を次に示します。

**(a) 動的登録の場合**

```
tslnkrm -a HiRDB_DB_SERVER_1 -s pdtxa_switch
-o 'C:¥win32app¥hitachi¥hirdb_s¥client¥lib¥pdcltxm.lib' -r -m
tslnkrm -a HiRDB_DB_SERVER_2 -s pdtxa_switch
-o 'C:¥win32app¥hitachi¥hirdb_s¥client¥lib¥pdcltxm.lib' -r -m
```

〔説明〕

-a：RM 名を指定します。

-s：RM スイッチ名（XA スイッチ構造体の名称）を指定します。RM スイッチ名は，登録方法（動的登録又は静的登録）によって異なります。

-o：RM 関連オブジェクト名（共用ライブラリのファイル名）を指定します。

-r：動的登録する場合に指定します。

-m：OTS のデーモンがマルチスレッドで動作するようになります。

**(b) 静的登録の場合**

```
tslnkrm -a HiRDB_DB_SERVER_1 -s pdtxa_switch_y
-o 'C:¥win32app¥hitachi¥hirdb_s¥client¥lib¥pdcltxm.lib' -r -m
tslnkrm -a HiRDB_DB_SERVER_2 -s pdtxa_switch_y
-o 'C:¥win32app¥hitachi¥hirdb_s¥client¥lib¥pdcltxm.lib' -r -m
```

〔説明〕

-a：RM 名を指定します。

-s：RM スイッチ名（XA スイッチ構造体の名称）を指定します。RM スイッチ名は，登録方法（動的登録又は静的登録）によって異なります。

-o：RM 関連オブジェクト名（共用ライブラリのファイル名）を指定します。

-r：静的登録する場合に指定します。

-m：OTS のデーモンがマルチスレッドで動作するようになります。

### (3) TUXEDO の場合

%TUXDIR%¥udataobj¥RM ファイルで HiRDB を TUXEDO に登録します。%TUXDIR% は，TUXEDO システム・ソフトウェアがあるディレクトリの絶対パス名を示しています。RM ファイルの指定例を次に示します。

**(a) 動的登録の場合**

```
HiRDB_DB_SERVER;pdtxa_switch;C:¥HiRDB¥client¥lib¥pdcltxs.lib
```

**(b) 静的登録の場合**

```
HiRDB_DB_SERVER;pdtxa_switch_y;C:¥HiRDB¥client¥lib¥pdcltxs.lib
```

### (4) WebLogic Server の場合

WebLogic Server の JDBC 接続プールの「ドライバクラス名」と「プロパティ」で HiRDB を登録します。指定例を次に示します。

**(a) 静的登録の場合**

```
ドライバクラス名：JP.co.Hitachi.soft.HiRDB.JDBC.JdbhXADataSource
プロパティ(key=value)：user=認可識別子
                    password=パスワード
                    XAOpenString=トランザクションデータソースで定義した名称
                    dataSourceName=JDBC接続プールで指定した名称
                    Description=トランザクションデータソースで定義した名称
```

〔説明〕

登録するときに指定する JDBC ドライバのパッケージ名称と Driver クラス名称を次に示します。

- パッケージ名称：JP.co.Hitachi.soft.HiRDB.JDBC
- Driver クラス名称：JdbhXADataSource

## 7.2.8 トランザクションマネジャへの登録の変更

トランザクションマネジャへの登録を変更する場合には，次に示す手順に従ってトランザクションマネジャに HiRDB を登録し直してください。なお，WebLogic Server の場合は，静的登録だけなので，登録の変更はできません。

### (1) OpenTP1 の場合

**〈手順〉**

1. OpenTP1 の trnlnkrm コマンドで，トランザクションマネジャに HiRDB を登録し直します。
2. OpenTP1 の trnmkobj コマンドで，トランザクション制御用オブジェクトファイルを再作成します。
3. 2 で再作成したトランザクション制御用オブジェクトファイル，及び「7.2.6　トランザクションマネジャに登録する情報」に示した情報を基に，HiRDB の XA ライブラリとリンクしていたすべての UAP を再リンケージしてください。再リンケージをしないと，UAP の動作を保証できません。

### (2) TPBroker for C++ の場合

**〈手順〉**

1. TPBroker for C++ の tslnkrm コマンドで，トランザクションマネジャに HiRDB を登録し直します。
2. TPBroker for C++ の tsmkobj コマンドで，トランザクション制御用オブジェクトファイルを再作成します。
3. 2 で再作成したトランザクション制御用オブジェクトファイル，及び「7.2.6　トランザクションマネジャに登録する情報」に示した情報を基に，HiRDB の XA ライブラリとリンクしていたすべての UAP を再リンケージしてください。再リンケージをしないと，UAP の動作を保証できません。

### (3) TUXEDO の場合

**〈手順〉**

1. %TUXDIR%¥udataobj¥RM でトランザクションマネジャに HiRDB を登録し直します。
2. TUXEDO の buildtms コマンドで，「7.2.6　トランザクションマネジャに登録する情報」に示した情報を基にトランザクションマネジャサーバのロードモジュールを再作成します。
3. TUXEDO の buildserver コマンドで，「7.2.6　トランザクションマネジャに登録する情報」に示した情報を基にサーバのロードモジュールを再作成します。
4. TUXEDO の buildclient コマンドで，「7.2.6　トランザクションマネジャに登録する情報」に示した情報を基にクライアントモジュールを再作成します。

## 7.2.9 トランザクションマネジャと HiRDB 間のコネクションが切断されたときの再接続方法

### (1) UAP で対処する方法

コネクションが切断された場合，実行中の UAP を終了後，再起動してください。再起動すると，自動的にコネクションが再接続されます。

UAP を再起動したくない場合は，コネクションが切断されたことを示すエラーが UAP に返ったときに，tx_open 関数を再発行してください。そうすれば，UAP を終了しなくてもサービスを続行できます。tx_open 関数を再発行するときのコーディング例を次に示します。

**コーディング例**

```
int connection = 1;
void service(char *in_data,long *in_len,char *out_data,long *out_len) {
    if (connection == 0) {
       tx_close();
       tx_open();            ……コネクション切断時のtx_open再発行処理
    }
    tx_begin();
    EXEC SQL INSERT INTO ......;                ……… SQL文発行
    if (SQLCODE == 0) {
        tx_commit();
        *out_data = "OK";
    } else {
        tx_rollback();
        *out_data = "NG";
        if (SQLCODE == -563 || SQLCODE == -722) {
            connection = 0;              ……コネクション切断を記憶
        }
    }
}
```

### (2) 連携する OLTP 製品が TPBroker for C++，及び WebLogic Server の場合

トランザクションの開始又は終了時に HiRDB とのコネクションを確立又は切断するため，途中で切断した場合も次回のトランザクション開始時にコネクションが再接続されます。

### (3) OpenTP1 の機能を使用する

動的登録の場合は，OpenTP1/Server Base の trn_rm_open_close_scope オペランドに transaction を指定してください。そうすれば，OpenTP1/Server Base はトランザクションの開始又は終了で HiRDB とのコネクションを確立又は切断します。したがって，途中でコネクションが切断されても，次回のトランザクションの開始時にコネクションが再接続されます。

静的登録の場合は，トランザクションの開始時に HiRDB とのコネクションが確立されているかどうかをトランザクションマネジャが確認します。コネクションが切断されている場合は，自動的に再接続されて，トランザクションを開始します。

### (4) HiRDB の XA インタフェースに対応したクライアントライブラリの再接続

トランザクションマネジャでトランザクションを開始して，最初に HiRDB にアクセスする SQL 文を実行するまでに，HiRDB とのコネクションが切断されていた場合，SQL 文の実行時に HiRDB クライアントライブラリでコネクションが再接続されます。ただし，ネットワーク障害による切断は検知できないため，再接続されません。

## 7.2.10 注意事項

### (1) SQL 関連の注意事項

1. リソースマネジャへの接続や切断を行う権限は，トランザクションマネジャにあります。UAP にリソースマネジャへの接続又は切断を行う SQL 文を記述しないでください。また，トランザクションの進行を調整し監視する権限も，トランザクションマネジャにあります。UAP にトランザクションをロールバック又はコミットする SQL 文を記述しないでください。したがって，EXEC SQL COMMIT WORK 文，EXEC COMMIT WORK RELEASE 文などもエラーになります。
2. 定義系 SQL 文はエラーとなります。CREATE TABLE などの定義系 SQL 文は自動的にコミットを指示するため，UAP に定義系 SQL 文を記述しないでください。

### (2) マルチスレッド用のライブラリに関する注意事項

一つのトランザクションから HiRDB サーバに対して複数のスレッドを使用して別々に接続できません。マルチスレッド環境であってもトランザクションから接続する HiRDB サーバのサーバプロセスは一つです。したがって，一つのトランザクションから同時に実行できるスレッドは一つであり，同一トランザクション内で複数のスレッドを使用して SQL 文を同時に実行できません。

### (3) 一相最適化に関する注意事項

HiRDB はトランザクションマネジャがサポートしている一相最適化に対応しています。トランザクションマネジャはトランザクションブランチによって変更した共有リソースが HiRDB だけの場合，トランザクションブランチに対して一相コミットを要求できます。トランザクションマネジャが一相最適化を使用して一相コミットを要求してきた場合，HiRDB はトランザクションブランチの結果を決めた後，トランザクションブランチの情報を削除してトランザクションマネジャへ応答を返します。

一相最適化を使用しているトランザクションマネジャでは，グローバルトランザクションに関して安定ストレージに記憶する必要もなく，何らかの障害が発生してもその結果を知る必要もありません。したがって，次に示す条件をすべて満たす場合，トランザクションの完了種別がトランザクションマネジャと HiRDB で一致しないことがあります。

- 一相最適化を使用するトランザクションマネジャと XA インタフェースを使用して接続している
- 更新系トランザクションのコミットメント制御をトランザクションマネジャが一相最適化している
- コミットメント処理中にトランザクションマネジャの UAP が異常終了する

これらの条件下では，HiRDB のトランザクションブランチの結果を，トランザクションマネジャが発生した障害の結果として知ることができません。そのため，トランザクションの完了種別がトランザクションマネジャと HiRDB で一致しないことがあります。

これを防ぐには，更新系トランザクションのコミットメント制御をする場合，トランザクションマネジャの一相最適化を使用しないでください。

### (4) WebLogic Server 使用時の注意事項

- コミット又はロールバックは前処理，カーソルを削除します。再度前処理から実行してください。
- ResultSet オブジェクトでカーソルをオープンしたままコミット又はロールバックをすると，HiRDB のトランザクションが決着しないで残ることがあります。ResultSet オブジェクトを使用する場合は，コミット又はロールバックの前に必ず ResultSet.close() メソッドを実行してカーソルをクローズしてください。

### (5) 高速系切り替え機能を使用している場合の注意事項

次に示す条件をすべて満たす場合は注意が必要です。

- HiRDB/ パラレルサーバの場合はシステムマネジャがあるユニットを高速系切り替え機能の対象にしている
- X/Open に従った API を使用した OLTP 製品（OpenTP1 又は TPBroker for C++ など）と連携している
- HiRDB クライアントのバージョンが 06-02-/A 以前である
- OLTP 製品のクライアント環境変数 PDHOST に指定している現用系が待機系として待機完了状態になっている

この場合，OLTP 製品が未決着トランザクションの回復処理をすると，X/Open に従った API がエラーリターンしてトランザクションが回復されないことがあります。この現象が発生する場合は，HiRDB クライアントのバージョンを 06-02-/B 以降にバージョンアップしてください。業務を停止させたくないなどの理由で HiRDB クライアントのバージョンアップがすぐにできない場合は，現用系の HiRDB（ユニット）を待機系から実行系に系を切り替えてください。ただし，これは一時的な対応策です。HiRDB クライアントのバージョンアップで対応してください。

# 7.3 HiRDB External Data Access 機能との連携

HiRDB External Data Access 機能を使用すると，他社製品を含む異機種の DBMS で構築した複数のデータベースの表に対して，HiRDB のインタフェースでアクセスできます。また，他種データベースが複数存在する場合でも，それらの情報を一つの表（外部表又はビュー表を使用します）として参照，更新できます。HiRDB External Data Access 機能を使用するには，HiRDB External Data Access が必要になります。

なお，HiRDB External Data Access 機能は，HiRDB/ パラレルサーバ限定の機能です。

## (1) HiRDB システム定義の指定

HiRDB システム定義で次に示すオペランド，及び定義文を指定します。

- pd_max_foreign_server オペランド
  接続する外部サーバの最大数を指定します。
- pdhubopt オペランド
  Hub 最適化情報定義を適用する外部サーバを指定します。
- pd_foreign_server_libpath オペランド
  外部サーバのクライアントライブラリのパス名を指定します。
- 外部サーバで必要な環境変数
  外部サーバで必要な環境変数をバックエンドサーバ定義で指定します。
- 外部サーバ情報定義
- Hub 最適化情報定義

前記のオペランドのほかに，次に示すオペランドの指定値を見直してください。

- pdstbegin オペランド
  このオペランドで，次に示す HiRDB External Data Access 機能関連の統計情報を取得するかどうかを指定します。
  - 外部サーバの稼働に関する統計情報
  - 外部サーバの利用状況に関する統計情報

## (2) 環境設定方法

HiRDB External Data Access 機能を使用する場合のシステムの環境設定及び運用方法については，マニュアル「HiRDB External Data Access Version 8」を参照してください。

# 7.4　JP1 との連携

JP1 と連携すると，HiRDB を含むシステム全体のイベントを一元管理したり，イベントを契機としてジョブを自動起動したりできます。HiRDB が連携する JP1 製品を次に示します。

- JP1/Base
- JP1/Integrated Management - Manager（JP1 Version 7i より前は JP1/Integrated Manager）
- JP1/Automatic Job Management System 2

以降，JP1/Integrated Management - Manager を JP1/IM，JP1/Automatic Job Management System 2 を JP1/AJS2 と表記します。

なお，Windows（x64）の HiRDB は，JP1 と連携できません。

## 7.4.1　JP1/Base へのイベント通知

HiRDB の開始，終了などのイベントを，JP1 のイベントを管理している JP1/Base に通知できます。通知した HiRDB のイベントは JP1 イベントとして JP1/Base が管理します。これによって，JP1/IM でイベントを管理したり，JP1/AJS2 と連携してジョブを自動実行したりできるようになります。JP1/IM によるイベント監視については「7.4.2　JP1/IM によるイベント管理」を，JP1/AJS2 との連携によるジョブの自動実行については「7.4.3　JP1/AJS2 との連携によるジョブの自動実行」を参照してください。

JP1/Base については，マニュアル「JP1/Base 運用ガイド」を参照してください。

### (1) イベント通知の方法

HiRDB のイベントを JP1/Base に通知するには，次のようにオペランドを指定します。

- pd_jp1_use オペランド：Y
- pd_jp1_event_level オペランド：1 又は 2

pd_jp1_event_level オペランドに 1 を指定すると，**基本属性**だけ通知します。2 を指定すると，**拡張属性**も通知します。

### (2) 通知できる HiRDB のイベント

JP1/Base に通知できる HiRDB のイベントを次の表に示します。

**表 7-5　JP1/Base に通知できる HiRDB のイベント**

| イベント | イベント ID ※1 | メッセージ | 詳細情報※2 | 登録するタイミング | pd_jp1_event_level の値 | |
|---|---|---|---|---|---|---|
| | | | | | 1 | 2 |
| 開始 | 0x00010C00<br>0x00010C80 | KFPS05210-I 又は PS05219-I ※3 | "start" | システム起動完了メッセージ（KFPS05210-I 又は KFPS05219-I）出力後 | ○ | ○ |

| イベント | イベント ID※1 | メッセージ | 詳細情報※2 | 登録するタイミング | pd_jp1_event_level の値 | |
|---|---|---|---|---|---|---|
| | | | | | 1 | 2 |
| 正常終了 | 0x00010C01<br>0x00010C81 | KFPS01850-I※3 | "end_normal" | HiRDB のシステム終了後（HiRDB 停止完了直前又は KFPS01850-I メッセージ出力直後） | ○ | ○ |
| 計画停止 | 0x00010C01<br>0x00010C81 | KFPS01850-I※3 | "end_plan" | | ○ | ○ |
| 強制終了 | 0x00010C01<br>0x00010C81 | KFPS01850-I※3 | "end_force" | | ○ | ○ |
| メッセージログ出力 | 0x00010C03<br>0x00010C83 | HiRDB メッセージ | NULL | メッセージログ出力後 | ○※4 | ○※4 |
| 異常終了 | 0x00010C02<br>0x00010C82 | KFPS01821-E | NULL | • HiRDB の異常終了時<br>• サーバモードで系切り替えを使用する場合はクラスタソフトウェアへの異常終了連絡後 | × | ○ |
| 定義変更 | 0x00010C04<br>0x00010C84 | KFPS04666-I | NULL | HiRDB のシステム正常起動完了時 | × | ○ |
| RD エリア満杯 | 0x00010C05<br>0x00010C85 | KFPH00213-W | NULL | RD エリア満杯エラーメッセージ（KFPH00213-W）出力時 | × | ○ |
| RD エリア拡張不可 | 0x00010C06<br>0x00010C86 | KFPX14229-E | NULL | pdmod（expand 指定）実行エラー時 | × | ○ |
| ログファイル空き領域警告 | 0x00010C07<br>0x00010C87 | KFPS01160-E 又は KFPS01162-W | NULL | システムログファイル空き領域警告メッセージ出力時 | × | ○ |

（凡例）

NULL：空です。何も通知されません。

○：このイベントを通知します。

×：このイベントを通知しません。

注※1

上段は HiRDB/ シングルサーバの場合を示し，下段は HiRDB/ パラレルサーバの場合を示しています。

注※2

詳細情報のデータはテキスト形式です。データの形式を次に示します。

ssss　　mm…mm　¥0

ssss　：システム識別子（0 ～ 4 バイト）

mm…mm：詳細情報欄で示した文字列

注※3

pd_jp1_event_level=1 の場合は NULL になります。

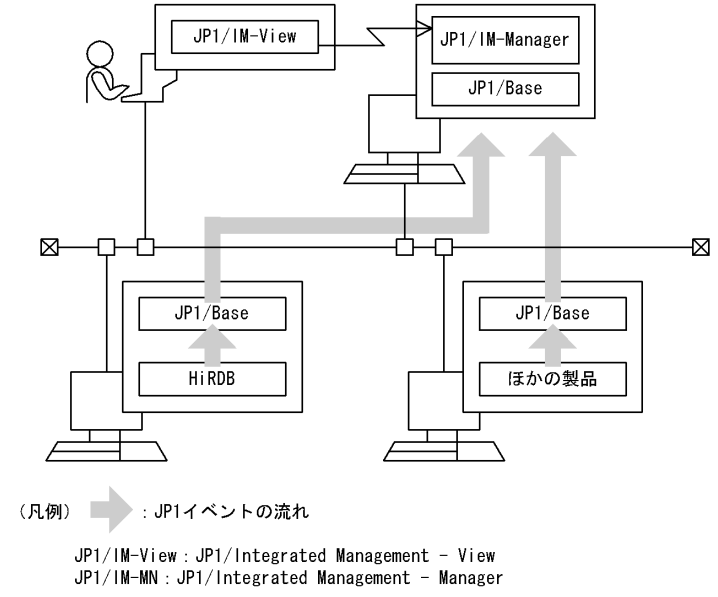
注※4

pd_jp1_event_msg_out=N の場合は通知しません。

## 7.4.2　JP1/IM によるイベント管理

JP1/IM は，JP1/Base が管理する JP1 イベントを最適化（フィルタリング）し，システムで発生した事象を JP1 イベントによって一元管理します。JP1/Base に HiRDB のイベントを通知していると，ほかの製品のイベントと同様に JP1/IM で管理できます。ユーザは JP1/IM が提供する画面でイベントを確認できます。JP1/IM によるイベント監視の概要を次の図に示します。

**図 7-7　JP1/IM によるイベント監視の概要**




JP1/Integrated Management によるイベント監視の概要については，JP1 のバージョンに応じて，該当するマニュアルを参照してください。

- JP1 Version 8 **の場合**
  「JP1/Integrated Management - Manager システム構築・運用ガイド」
- JP1 Version 7i **の場合**
  「JP1 Version 7i JP1/Integrated Manager - Console システム構築・運用ガイド」

### （1）JP1/IM で HiRDB 固有の拡張属性を表示するための準備

JP1/IM で HiRDB 固有の拡張属性を表示する場合，HiRDB が提供する**イベント属性定義ファイル**を次のディレクトリにコピーする必要があります。

- JP1/Integrated Management - Manager（JP1 Version 7i より前は JP1/Integrated Manager - Central Console）のインストールディレクトリ ¥conf¥console¥attribute

イベント属性定義ファイルは，HiRDB のインストールディレクトリ下の sample ディレクトリに格納されます。イベント属性定義ファイル名称は，HITACHI_HIRDB_NT_attr_ja.conf です。

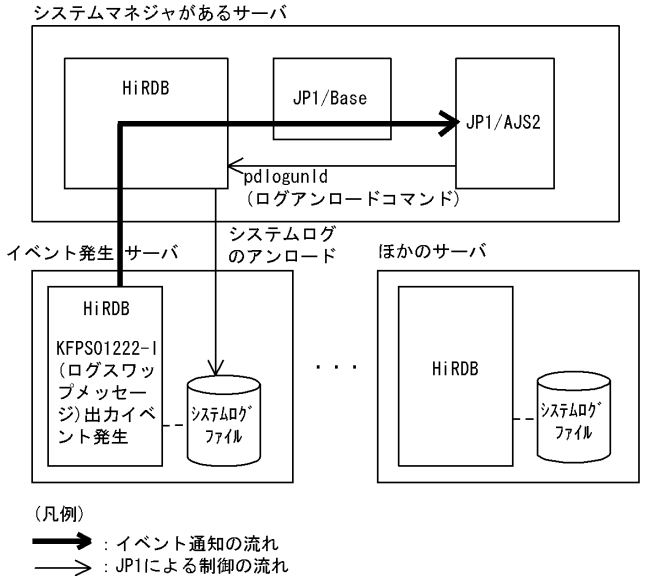
なお，マルチ HiRDB の場合，稼働する HiRDB の中で最新のバージョンのイベント属性定義ファイルをコピーしてください。

## 7.4.3 JP1/AJS2 との連携によるジョブの自動実行

HiRDB/ パラレルサーバの場合，各サーバのシステムログファイルのアンロードなどの運用が煩雑になることがあります。このような場合，JP1/Base に HiRDB のイベントを通知していると，通知したイベントを契機として，JP1/AJS2 でジョブを自動実行し，HiRDB の運用が自動化できるようになります。

JP1 との連携によるシステムログファイルのアンロード時の自動制御を次の図に示します。

**図 7-8 JP1 との連携によるシステムログファイルのアンロード時の自動制御**




JP1/AJS2 については，JP1 のマニュアルを参照してください。ただし，JP1 のバージョンによってマニュアル名が異なります。該当するバージョンに対応するマニュアルを参照してください。

- JP1 Version 7i **以降の場合**
  「JP1/Automatic Job Management System 2 解説」
- JP1 Version 6 **の場合**
  「JP1/Automatic Job Management System 2 運用・操作編」

# 8 HiRDB/シングルサーバの設計

この章では，HiRDB/ シングルサーバのシステム構成，HiRDB ファイルシステム領域の設計方法，システムファイルの設計方法及び RD エリアの配置方法の考慮点について説明します。


8.1 HiRDB/ シングルサーバのシステム設計

8.2 HiRDB ファイルシステム領域の設計

8.3 システムファイルの設計

8.4 RD エリアの配置

# 8.1 HiRDB/ シングルサーバのシステム設計

ここでは，HiRDB/ シングルサーバのシステム設計とシステム構成について説明します。

## 8.1.1 システム設計

### (1) HiRDB/ シングルサーバが使用するメモリ

HiRDB/ シングルサーバが使用するメモリについて説明します。

HiRDB/ シングルサーバは次に示すメモリを使用します。

- 共用メモリ
- プロセス固有メモリ

#### (a) メモリ所要量

HiRDB/ シングルサーバが必要とするメモリ所要量を見積もってください。HiRDB/ シングルサーバのメモリ所要量については，「15.1 HiRDB/ シングルサーバのメモリ所要量の見積もり」を参照してください。

#### (b) 共用メモリのページ固定

HiRDB では，次に示す共用メモリを実メモリ上に固定できます。

- ユニットコントローラ用共用メモリ
- グローバルバッファ用共用メモリ
- 動的変更したグローバルバッファが使用する共用メモリ
- インメモリデータバッファ用共用メモリ

共用メモリを実メモリ上に固定すると，ページの入出力が少なくなるため，性能が安定します。

**前提条件**

共用メモリのページ固定をするための前提条件を次に示します。

Windows のバージョン

Windows では，Windows Server 2003 Service Pack 1 以降でサポートされている Large Page の機能を使用してページ固定をします。そのため，Large Page がサポートされているバージョンの Windows であることが前提となります。Windows がページ固定に対応しているかどうかは，pdntenv -os コマンドコマンドで確認してください。

**ページロックの権限**

共用メモリのページ固定をする場合，メモリ内のページをロックできる権限が必要です。サービス実行時に使用するログオンのアカウントによって，この権限の有無が異なります。権限の設定方法を次に示します。

| サービス実行時に使用するログオンのアカウント | 権限の設定方法 |
|---|---|
| HiRDB 管理者 | HiRDB 管理者を，OS の［ローカル セキュリティ設定］－［ローカル ポリシー］－［ユーザー権利の割り当て］で，「メモリ内のページのロック」に設定してください。 |
| ローカルシステムアカウント | ローカルシステムアカウントにはメモリ内のページをロックできる権限があります。そのため，権限の設定は必要ありません。 |

**動作環境の設定**

共用メモリのページ固定をするには，共用メモリの割り当て先の指定が必要です。pdntenv コマンドの -shmfile オプションで page を指定し，共用メモリの割り当て先をページングファイル（仮想メモリ）に設定してください。

**ページ固定の方法**

共用メモリのページ固定の方法について，共用メモリの種別ごとに説明します。

- ユニットコントローラ用共用メモリ
  システム共通定義，又はユニット制御情報定義の pd_shmpool_attribute オペランドに fixed を指定します。
- グローバルバッファ用共用メモリ
  システム共通定義，又はユニット制御情報定義の pd_dbbuff_attribute オペランドに fixed を指定します。
- 動的変更したグローバルバッファが使用する共用メモリ
  システム共通定義，又はユニット制御情報定義の pd_dbbuff_attribute オペランドに fixed を指定します。これによって，pdbufmod コマンドを実行して動的変更したグローバルバッファが使用する共用メモリも実メモリ上に固定されます。
- インメモリデータバッファ用共用メモリ
  pdmemdb コマンドの -p オプションに fixed を指定します。

**注意事項**

実メモリ上に連続した領域が確保できなかった場合，共用メモリのページ固定ができません。ページ固定に失敗した場合の HiRDB の動作を次に示します。

**ユニットコントローラ用，又はグローバルバッファ用共用メモリの場合**
ページ固定をしないで共用メモリを確保し処理を続行します。

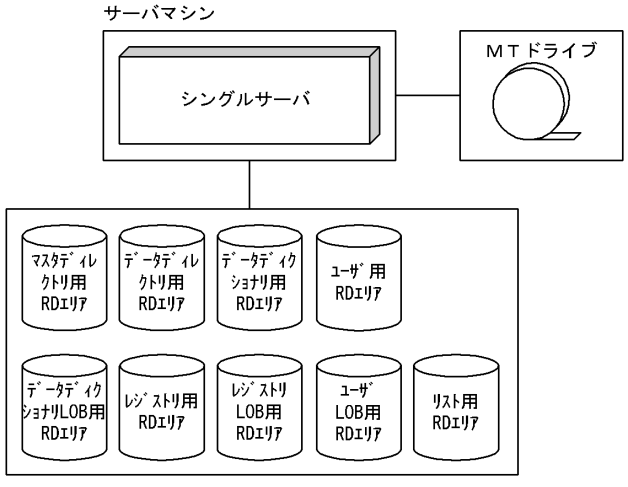
**動的変更したグローバルバッファが使用する共用メモリ，又はインメモリデータバッファ用共用メモリの場合**
HiRDB，又はコマンドが異常終了します。

## 8.1.2 HiRDB/シングルサーバのシステム構成

HiRDB/ シングルサーバのシステム構成例を次の図に示します。

HiRDB/ シングルサーバのシステム構成は，HiRDB システム定義で定義します。HiRDB システム定義の定義例については，マニュアル「HiRDB Version 8 システム定義」を参照してください。

**図** 8-1 HiRDB/**シングルサーバのシステム構成**



# 8.2 HiRDB ファイルシステム領域の設計

HiRDB のシステム構築時に，HiRDB ファイルを作成する HiRDB ファイルシステム領域を作成します。ここでは，HiRDB ファイルシステム領域を作成するときの設計方針について説明します。

HiRDB ファイルシステム領域は，次に示す用途ごとに作成することをお勧めします。これによって，用途やアクセス特性の異なるファイルへの入出力が競合することを回避できます。

- RD エリア用
- システムファイル用
- 作業表用ファイル用
- ユティリティ用
- リスト用 RD エリア用

## 8.2.1 RD エリア用の HiRDB ファイルシステム領域の設計

RD エリア用の HiRDB ファイルシステム領域の設計方針について説明します。

### (1) 信頼性向上のための方針

特にありません。

### (2) 性能向上のための方針

1. RD エリアを作成する HiRDB ファイルシステム領域は，次に示す RD エリア用ごとに作成することをお勧めします。
   - システム用 RD エリア
   - データディクショナリ LOB 用 RD エリア
   - ユーザ用 RD エリア
   - ユーザ LOB 用 RD エリア
   - レジストリ用 RD エリア
   - レジストリ LOB 用 RD エリア
2. システムファイル用の HiRDB ファイルシステム領域と，RD エリア用の HiRDB ファイルシステム領域は，別々のハードディスクに作成することをお勧めします。これによって，シンクポイントダンプを取得するときに入出力の分散ができ，シンクポイントダンプの取得処理時間を短縮できます。

## 8.2.2 システムファイル用の HiRDB ファイルシステム領域の設計

システムファイル用の HiRDB ファイルシステム領域の設計方針について説明します。

### (1) 信頼性向上のための方針

1. システムファイル用の HiRDB ファイルシステム領域は二つ以上作成してください。一つしか作成しないと，システムファイルがあるハードディスクに障害が発生した場合，HiRDB が稼働できなくなります。
2. システムファイル用の HiRDB ファイルシステム領域は別々のハードディスクに作成してください。そうすれば，どちらかのハードディスクに障害が発生しても，HiRDB を再開始できます。

### (2) 性能向上のための方針

システムファイル用の HiRDB ファイルシステム領域と，RD エリア用の HiRDB ファイルシステム領域

は，別々のハードディスクに作成することをお勧めします。これによって，シンクポイントダンプを取得するときに入出力の分散ができ，シンクポイントダンプの取得処理時間を短縮できます。

## 8.2.3 作業表用ファイル用の HiRDB ファイルシステム領域の設計

作業表用ファイル用の HiRDB ファイルシステム領域の設計方針について説明します。

### (1) 設計方針

作業表用ファイル用の HiRDB ファイルシステム領域長は，その HiRDB ファイルシステム領域に作成する作業表用ファイルの総容量より大きくしてください。なお，pdfmkfs コマンドで -a オプションを指定すると，HiRDB ファイルシステム領域を自動的に拡張できます。作業表用ファイルの総容量が HiRDB ファイルシステム領域長に達した場合，自動で HiRDB ファイルシステム領域を拡張できるため，-a オプションを指定することをお勧めします。※

作業表用ファイルの容量については，「18. 作業表用ファイルの容量の見積もり」を参照してください。

注※

HiRDB 再開始時に作業表用ファイル用の HiRDB ファイルシステム領域のディスク占有サイズを削減したい場合は，HiRDB 再開始前に pdfmkfs コマンドを実行して作業表用ファイル用の HiRDB ファイルシステム領域を再初期化してください。

### (2) 最大使用量を調べる方法

作業表用ファイル用の HiRDB ファイルシステム領域の最大使用量を次に示す方法で調べられます。

pdfstatfs -d **作業表用の HiRDB ファイルシステム領域名**

-d：

HiRDB ファイルシステム領域の最大使用量を表示するオプションです。出力情報の peak capacity が最大使用量です。なお，上記の最大使用量は pdfstatfs コマンドでクリアできます。

pdfstatfs -c **作業表用の HiRDB ファイルシステム領域名**

-c：

HiRDB ファイルシステム領域の最大使用量をクリアするオプションです。

## 8.2.4 ユティリティ用の HiRDB ファイルシステム領域の設計

ユティリティ用（バックアップファイル，アンロードデータファイル，アンロードログファイル作成用）の HiRDB ファイルシステム領域の設計方針について説明します。ユティリティ用の HiRDB ファイルシステム領域には，次に示すファイルを作成します。

- バックアップファイル
- アンロードデータファイル
- アンロードログファイル
- 差分バックアップ管理ファイル

### (1) 設計方針

1. バックアップファイル作成用にする場合は，HiRDB ファイルシステム領域長をバックアップ対象 RD エリアの総容量より大きくしてください。RD エリアの容量については，「16. RD エリアの容量の見積もり」を参照してください。
2. 系切り替え機能使用時は，アンロードログファイルを共有ディスク上に作成してください。

3. アンロードログファイル作成用にする場合は，pdfmkfs コマンドのオプションに次に示す値を指定してください。
   - -k オプション：使用目的には UTL（ユティリティ用の HiRDB ファイルシステム領域）を指定します。
   - -n オプション：HiRDB ファイルシステム領域長には次に示す計算式の値を指定します。

     **（アンロードするシステムログファイルの総レコード数※1×システムログファイルのレコード長）※2×作成するアンロードログファイル数×** 1.2 ÷ 1048576
   - -l オプション：最大ファイル数には，作成するアンロードログファイル数を指定します。
   - -e オプション：最大増分回数には，作成するアンロードログファイル数× 24 を指定します。

   注※ 1

   システムログファイルの自動拡張機能を適用している場合，pd_log_auto_expand_size オペランドの拡張上限サイズに指定した値で計算してください。

   注※ 2

   システムログファイルの概算値です。

### (2) 最大使用量を調べる方法

ユティリティ用の HiRDB ファイルシステム領域の最大使用量を次に示す方法で調べられます。

pdfstatfs -d **ユティリティ用の** HiRDB **ファイルシステム領域名**

-d：

HiRDB ファイルシステム領域の最大使用量を表示するオプションです。出力情報の peak capacity が最大使用量です。なお，上記の最大使用量は pdfstatfs コマンドでクリアできます。

pdfstatfs -c **ユティリティ用の** HiRDB **ファイルシステム領域名**

-c：

HiRDB ファイルシステム領域の最大使用量をクリアするオプションです。

## 8.2.5 リスト用 RD エリア用の HiRDB ファイルシステム領域の設計

リスト用 RD エリア用の HiRDB ファイルシステム領域の設計方針について説明します。

### (1) 設計方針

リストは検索の一時的な中間結果を保存するものなので，ほかの RD エリアほど信頼性は要求されません。

### (2) 性能向上のための方針

リスト用 RD エリア用の HiRDB ファイルシステム領域は，次に示す HiRDB ファイルシステム領域とは別々のハードディスクに作成することをお勧めします。別々のハードディスクに作成すると，リストを検索するときに入出力を分散できるため，処理時間を短縮できます。

- ユーザ用 RD エリア用の HiRDB ファイルシステム領域
- ユーザ LOB 用 RD エリア用の HiRDB ファイルシステム領域
- 作業表用ファイル用の HiRDB ファイルシステム領域

## 8.2.6 HiRDB ファイルシステム領域の最大長

HiRDB ファイルシステム領域の最大長を次の表に示します。

表 8-1 HiRDB ファイルシステム領域の最大長

| 条件 | HiRDB ファイルシステム領域の最大長 |
|---|---|
| pd_large_file_use ＝ N 指定時 | 2,047 メガバイト |
| pd_large_file_use=Y 指定時（省略値） | 1,048,575 メガバイト |

**注意事項**

HiRDB ファイルシステム領域長が 100 ギガバイトを超える場合は，pd_ntfs_cache_disable=Y を指定してください。pd_ntfs_cache_disable=Y を指定する場合は，LOB 用 RD エリアとそのほかの RD エリアを同じ HiRDB ファイルシステム領域に格納しないようにしてください。

## 8.2.7 系切り替え機能使用時の注意事項

系切り替え機能で使用する HiRDB ファイルシステム領域を作成する場合の注意事項を次に示します。

- pdfmkfs コマンドの -i オプションを指定してください。-i オプションを指定しないと，HiRDB ファイルシステム領域の拡張とサーバマシンの電源異常が同時に発生した場合にファイルが破壊されることがあります。
- pdfmkfs コマンドの -k オプションに SVR（省略値）を指定しないでください。-k オプションに SVR を指定して作成した HiRDB ファイルシステム領域に対して，HiRDB は Windows のファイルキャッシュに書き込んだ時点で処理を続行します。このため，ディスク上にデータが書き込まれる前に異常が発生するとファイルの整合性がとれなくなります。

# 8.3 システムファイルの設計

ここでは，システムファイルの設計方針について説明します。

## 8.3.1 システムログファイルの設計

システムログファイルのスワップ時に，スワップ先にできる状態のシステムログファイルがないとHiRDB は異常終了します。これを予防するため，HiRDB にはシステムログファイルの空き容量を監視する機能（**システムログファイルの空き容量監視機能**）があります。この機能は，システムログファイルの空き率が**警告値**未満になったときに作動します。次に示す二つのレベルのどちらかを選択できます。

**レベル 1：**

システムログファイルの空き率が警告値未満になった場合，警告メッセージ KFPS01162-W を出力します。

**レベル 2：**

システムログファイルの空き率が警告値未満になった場合，新規トランザクションのスケジューリングを抑止して，サーバ内の全トランザクションを強制終了します。このとき，KFPS01160-E メッセージを出力します。これによって，システムログの出力量を抑えます。

レベル 2 を選択した場合，システムログファイルの空き容量が不足したときにサーバ内の全トランザクションが強制終了されます。このため，システムログファイルの設計をより正確に行う必要があります。

システムログファイルの設計方針について説明します。

### (1) 設計方針

1. すべてのシステムログファイルのレコード長及びレコード数を同じにしてください。
2. 作成できるシステムログファイルは，2 ～ 200 グループです（ただし，6 グループ以上作成することを推奨します）。
   システムログファイルの容量不足によって異常終了した HiRDB を再開始する場合，作成済みのシステムログファイルと同じ数のシステムログファイルを追加する必要があります。例えば，最大量（100 ギガバイト）のシステムログファイルを 50 個で 50 グループ作成している場合は，更に最大量のシステムログファイルを 50 個で 50 グループ追加します。したがって，作成するシステムログファイルは 100 グループ以内とすることをお勧めします。
3. システムログファイルの総容量の上限は 20 テラバイトになります。
4. アンロードする回数を少なくしたい場合は，システムログファイルの 1 ファイル容量を大きくします。
5. HiRDB 運用ディレクトリがあるディスクに大量に入出力が発生するファイル（システムログのアンロードログファイルなど）を作成しないでください。
6. 一つのシステムログファイルの容量は，次に示す条件を満たす必要があります。

一つのシステムログファイルの容量（バイト）≧↑（a ＋ 368）÷ c ↑× c × b × d

a：pd_log_max_data_size オペランドの値

b：pd_log_sdinterval オペランドの値

c：pd_log_rec_leng オペランドの値

d：pd_spd_assurance_count オペランドの値

7. 全システムログファイルの容量（二重化している場合は片系の容量だけを対象とする）は，次に示す二

つの条件を満たす必要があります。

**条件** 1

「17.1.1(1) システムログファイルの総容量の求め方」で見積もった容量以上の値にしてください。

**条件** 2

長大トランザクションが終了するまでは，長大トランザクション開始以降のシステムログファイルの上書きができません。また，シンクポイントダンプファイルの有効保証世代とカレント世代のシステムログファイルは上書きできません。そのため，これらの分のシステムログファイル容量を確保してください。次の計算式で求めます。

システムログファイルの総容量（バイト）≧3 × a ×（b ＋ 1）

**a：**

データベースを更新するトランザクションのうち，実行時間が最大のトランザクション実行中に該当するサーバで出力されることのあるシステムログ量

システムログ量の見積もり式については，「17.1　システムログファイルの容量の見積もり」を参照してください。

**b：**pd_spd_assurance_count オペランドの値

シンクポイントダンプファイルの有効保証世代数です。

### (a) システムログファイルの世代数と運用への影響

システムログファイルの総容量が同じ場合，システムログファイルの世代数によって，1 世代当たりの容量が変化します。システムログファイルの世代数と運用への影響を次の表に示します。システムログファイルの総容量は同じとします。

**表** 8-2　**システムログファイルの世代数と運用への影響**

| 比較項目 | システムログファイルの構成 | |
|---|---|---|
| | 世代数を少なくした場合 | 世代数を多くした場合 |
| システムログファイル 1 世代当たりの容量 | 大きくなります。 | 小さくなります。 |
| スワップ間隔 | システムログファイル 1 世代当たりの容量が大きくなるため，スワップ間隔は長くなります。 | システムログファイル 1 世代当たりの容量が小さくなるため，スワップ間隔は短くなります。 |
| アンロード回数 | スワップ間隔が長くなるため，アンロード回数は少なくなります。 | スワップ間隔が短くなるため，アンロード回数は多くなります。 |
| ディスク障害などでシステムログファイルの数世代が使用できなくなった場合のシステムログ容量への影響 | • システムログファイル 1 世代当たりの容量が大きくなるため，ディスク障害時にデータベースの回復に用いるログ量は多くなり，データベース回復に掛かる時間は長くなります。<br>• システムログ容量の減少幅が大きいため，システムログの容量が減少したことによる HiRDB の稼働に及ぼす影響は大きくなります。 | • システムログファイル 1 世代当たりの容量が小さくなるため，ディスク障害時にデータベースの回復に用いるログ量は少なくなり，データベース回復に掛かる時間は短くなります。<br>• システムログ容量の減少幅が小さいため，システムログの容量が減少したことによる HiRDB の稼働に及ぼす影響は小さくなります。 |

通常の運用では，システムログファイルの世代数を少なくした場合の方がスワップ間隔及びアンロード回数の点で利点があります。ただし，障害発生時には，世代数を多くした場合の方が障害の影響が小さくなります。

## (2) 信頼性向上のための方針

### (a) システムログファイルの二重化

システムログファイルを二重化すると，HiRDB は両方の系に同じシステムログを取得します。取得したシステムログを読み込むとき，片方のファイルに異常が発生しても，もう一方のファイルからシステムログを読み込めるため，システムの信頼性を向上できます。なお，二重化する場合，ディスクをミラー化して二重化するより，HiRDB 管理下で二重化することをお勧めします。システムログファイルを二重化する場合は，それぞれの系のファイルを別々のハードディスクに作成してください。

システムログファイルを二重化する場合は，サーバ定義で次に示すオペランドを指定してください。

- pd_log_dual = Y
- pdlogadpf オペランドの -b オプション（B 系のシステムログファイル名を指定します）

### (b) システムログファイルの片系運転

システムログファイルの片系運転は，システムログファイルを二重化する場合に適用されます。

システムログファイルに障害が発生して，両系とも使用できるシステムログファイルがない場合でも，HiRDB のユニットを異常終了しないで正常な系だけで処理を続行できます。これを**システムログファイルの片系運転**といいます。システムログファイルの片系運転をする場合は，サーバ定義で pd_log_singleoperation = Y を指定してください。

システムログファイルの片系運転に対し，両方のシステムログファイルで処理を続行すること（通常の処理形態）を**システムログファイルの両系運転**といいます。

### (c) システムログファイルの自動オープン

HiRDB を再開始するときに，上書きできる状態のシステムログファイルがない場合，予約のファイルがあれば HiRDB が予約のファイルをオープンして上書きできる状態にし，処理を続行します。これを**システムログファイルの自動オープン**といいます。

システムログファイルの自動オープンをする場合は，サーバ定義で pd_log_rerun_reserved_file_open = Y を指定してください。

## (3) システムログファイルのレコード長

システムログファイルのレコード長を pdloginit コマンドの -l オプションで指定します。レコード長は 1024，2048，4096 のどれかを選択できます。

### (a) 新規に HiRDB を構築する場合

新規に HiRDB を構築する場合は，レコード長を 1024 にすることをお勧めします。この場合，サーバ定義の pd_log_rec_leng オペランドで 1024 を指定してください。

### (b) 既に HiRDB を稼働している場合

レコード長が 1024 以外の場合，1024 に変更することをお勧めします。

システムログファイルのレコード長の変更方法については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

**レコード長を変更するときの考慮点**

- 全システムログファイルの総容量が変更になります。全システムログファイルの総容量の求め方については，「17.1.1　システムログファイルの総容量」を参照してください。
- ユニットコントローラが使用する共用メモリサイズが変更になります。ユニットコントローラが使

用する共用メモリサイズの求め方については，「15.1.3　ユニットコントローラが使用する共用メモリの計算式」を参照してください。

### (4) システムログファイルの定義

作成したシステムログファイルをどのファイルグループに対応させるかをサーバ定義の pdlogadfg 及び pdlogadpf オペランドで定義します。

## 8.3.2 シンクポイントダンプファイルの設計

シンクポイントダンプファイルの設計方針について説明します。

### (1) 設計方針

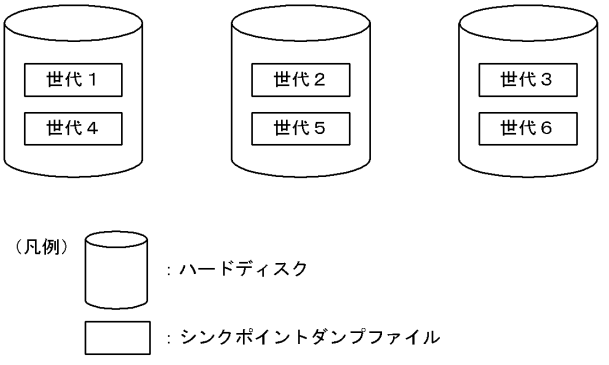
1. 作成できるシンクポイントダンプファイルは，2 ～ 60 グループです。pdlogadfg -d spd オペランドに ONL を指定した場合は，2 ～ 30 グループです。
2. HiRDB は pdlogadfg -d spd オペランドの指定順にシンクポイントダンプファイルを使用します。
3. 4 個以上のシンクポイントダンプファイルを作成することをお勧めします。
4. シンクポイントダンプファイルの容量が不足すると，HiRDB を開始できません。このため，シンクポイントダンプファイルのレコード数は，システム共通定義の最大同時接続数（pd_max_users オペランド）の指定値より大きな値にしてください。詳細なシンクポイントダンプファイルの容量計算式については，「17.2　シンクポイントダンプファイルの容量の見積もり」を参照してください。

### (2) 信頼性向上のための方針

#### (a) ファイル構成例

ハードディスク障害に備えて，シンクポイントダンプファイルをそれぞれ別々のハードディスクに作成してください。そのような構成を取れない場合は，隣り合わせの世代を別々のハードディスクに作成する構成にしてください。例を次の図に示します。

**図 8-2　隣り合わせの世代を別々のハードディスクに作成する構成例**




#### (b) シンクポイントダンプファイルの二重化

シンクポイントダンプファイルを二重化すると，HiRDB は A 系及び B 系の両方に同じシンクポイントダンプを取得します。取得したシンクポイントダンプを読み込むとき，片方のファイルに異常が発生しても，もう一方のファイルからシンクポイントダンプを読み込めるため，システムの信頼性を向上できます。また，二重化している場合，有効保証世代数を 1 世代にすると，信頼性を損ねることなく上書きできない状態のシンクポイントダンプファイル数を削減できます。

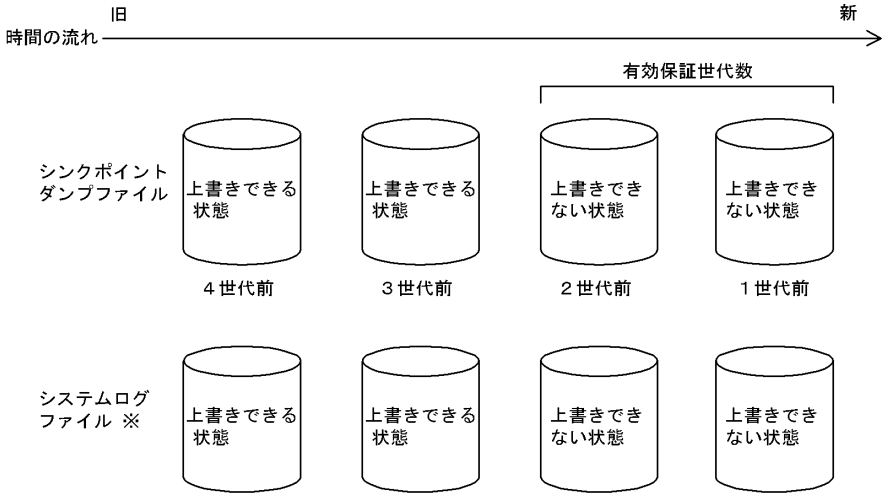
シンクポイントダンプファイルを二重化する場合は，サーバ定義で次に示すオペランドを指定してください。

- pd_spd_dual = Y
- pdlogadpf オペランドの -b オプション（B 系のシステムログファイル名を指定します）

### (c) シンクポイントダンプファイルの有効保証世代数

一つのシンクポイントダンプファイルには，HiRDB が 1 回に取得するシンクポイントダンプが格納されます。HiRDB は，シンクポイントダンプファイルを世代という概念で管理しています。HiRDB 管理者は，何世代前までのシンクポイントダンプファイルに対応するシステムログファイルを上書きできない状態にするかを指定できます。これを**シンクポイントダンプファイルの有効保証世代数**といいます。シンクポイントダンプファイルの有効保証世代数を次の図に示します。

**図 8-3 シンクポイントダンプファイルの有効保証世代数**




注※ シンクポイントダンプファイルに対応するシステムログファイルを示します。

〔説明〕

有効保証世代を 2 とすると，2 世代前までのシンクポイントダンプファイル及びそのシンクポイントダンプファイルに対応するシステムログファイルが，上書きできない状態になります。3 世代前より前の世代のシンクポイントダンプファイル及びそのシンクポイントダンプファイルに対応するシステムログファイルは，上書きできる状態になります。

シンクポイントダンプファイルの運用に必要なファイル数は，**有効保証世代数＋ 1** となります。シンクポイントダンプファイルの有効保証世代数は，サーバ定義の pd_spd_assurance_count オペランドで指定してください。

なお，シンクポイントダンプファイルを二重化している場合，必要な有効保証世代数は 1 世代をお勧めします。二重化していない場合，2 世代をお勧めします。

### (d) シンクポイントダンプファイルの縮退運転

シンクポイントダンプファイルに障害が発生して，運用に必要なファイル数（有効保証世代数＋ 1）を下回った場合でも，最低二つのファイルで処理を続行できます。これを**シンクポイントダンプファイルの縮退運転**といいます。

シンクポイントダンプファイルの縮退運転をする場合は，サーバ定義の pd_spd_reduced_mode オペランドを指定してください。

#### (e) シンクポイントダンプファイルの自動オープン

シンクポイントダンプファイルに障害が発生して，運用に必要なファイル数（有効保証世代数＋ 1）を下回った場合，予約のファイルがあれば HiRDB が予約のファイルをオープンして上書きできる状態にし，処理を続行します。これを**シンクポイントダンプファイルの自動オープン**といいます。

シンクポイントダンプの自動オープンをする場合は，サーバ定義で pd_spd_reserved_file_auto_open = Y を指定してください。

### (3) シンクポイントダンプファイルの定義

作成したシンクポイントダンプファイルをどのファイルグループに対応させるかを pdlogadfg 及び pdlogadpf オペランドで定義します。

なお，pdlogadfg オペランドだけを指定しておくと，HiRDB 稼働中にシンクポイントダンプファイルを追加できます。

## 8.3.3 ステータスファイルの設計

ステータスファイルの設計方針について説明します。

### (1) 設計方針

1. 両系のファイルに障害が起きないように，A 系と B 系のファイルは別々のディスクに作成します。
2. ステータスファイルの容量不足による HiRDB の異常終了を防ぐため，見積もったファイル容量よりも大きい容量の予備ファイルを幾つか作成してください。ステータスファイルは，容量が満杯になると予備ファイルとスワップしますが，満杯になったステータスファイルと予備ファイルの容量が同じ場合，スワップ先のファイルでも容量不足になり，HiRDB は異常終了します。そのため，例えばステータスファイルを 6 組作成する場合，そのうちの 2 組以上のファイル容量をほかのファイル容量より大きくすることをお勧めします。
3. A 系と B 系のファイルのレコード長及びレコード数を同じにしてください。
4. 作成できるユニット用ステータスファイルは 1 ～ 7 組です。
5. 作成できるサーバ用ステータスファイルは 1 ～ 7 組です。

### (2) 信頼性向上のための方針

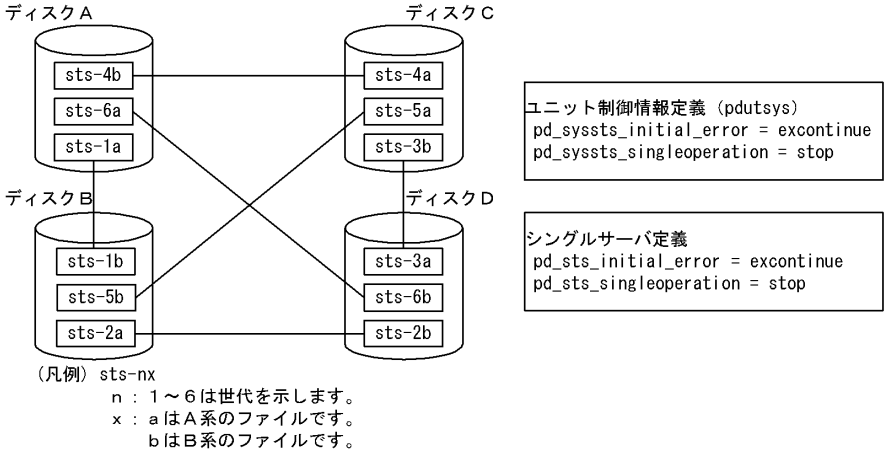
1. ステータスファイルは 3 組（二重化× 3=6 ファイル）以上用意し，ディスク障害によってすべてのステータスファイルが障害とならないように配置します。
2. 容量不足による HiRDB の異常終了を防ぐため，ステータスファイルの容量は見積もった容量の 1.2 倍以上の容量を準備することをお勧めします。
3. ステータスファイルには，HiRDB の再開始処理でシステムの状態を回復するために必要な情報を格納しています。予備ファイルがない状態で現用ファイルに障害が発生すると，システムの状態が回復できません。したがって，常に予備ファイルがあるように運用し，現用ファイルの障害に備えてください。

#### (a) お勧めする構成

ディスク障害発生時から回復時までの安全性を考えると，ステータスファイルは四つのディスクに 6 組（二重化× 6=12 ファイル）用意し，次の図のように配置することをお勧めします。また，片系運転中に正常な系に障害が発生すると，HiRDB を再開始できなくなるため，ステータスファイルの片系運転は適用しない（pd_syssts_singleoperation 及び pd_sts_singleoperation に stop を指定）ことをお勧めします。

四つのディスクにステータスファイルを 6 組配置する例を次の図に示します。

**図 8-4 四つのディスクに 6 組のステータスファイルを配置する例**




〔説明〕

このように配置すると，あるディスクで障害が発生した後に，更に別のディスクで障害が発生しても，残りの二つのディスクに両系とも正常なファイルが残るため，障害が発生していないディスクのステータスファイルを現用として HiRDB を稼働し続けることができます。例えば，ディスク A に障害が発生し，その後ディスク B にも障害が発生した場合でも，ディスク C 及び D にある両系のステータスファイル（sts-3a と sts-3b）を現用ファイルとして稼働し続けます。この状態で，更に現用ファイルの片系に障害が発生した場合，HiRDB は異常終了しますが，現用ファイルの片系ファイルが正常のため，障害が発生したディスクのどれか一つを回復すると HiRDB を再開始できます。

## (3) ステータスファイルの定義

pdstsinit コマンドで作成したステータスファイルをどの論理ファイルに対応させるかを pd_syssts_file_name_1 ～ 7 及び pd_sts_file_name_1 ～ 7 オペランドで定義します。

ユニット用ステータスファイルは，pd_syssts_file_name_1 ～ 7 オペランドで定義します。サーバ用ステータスファイルは，pd_sts_file_name_1 ～ 7 オペランドで定義します。

なお，pd_syssts_file_name_2 ～ 7 オペランド又は pd_sts_file_name_2 ～ 7 オペランドに，実体のないステータスファイルを定義しておくと，HiRDB 稼働中にステータスファイルを追加できます。ただし，この場合，次に示すオペランドを指定しておく必要があります。

**ユニット用ステータスファイルの場合**

- pd_syssts_initial_error
- pd_syssts_last_active_file

**サーバ用ステータスファイルの場合**

- pd_sts_initial_error
- pd_sts_last_active_file

## (4) ステータスファイルの片系運転

予備ファイルがない状況で現用ファイルの片系に障害が発生した場合，正常な系（片方の系）だけで処理

を続行することを**ステータスファイルの片系運転**といいます。ステータスファイルが片系運転になると，KFPS01044-I メッセージが出力されます。

片系運転中に現用ファイルに障害が発生すると，HiRDB を再開始できなくなるため，ステータスファイルの片系運転の適用は推奨しません。ステータスファイルの組数を増やし，予備ファイルがない状況が発生しないような運用をしてください。

なお，ステータスファイルの片系運転に対し，両方の系のステータスファイルで処理を続行すること（通常の処理形態）を**ステータスファイルの両系運転**といいます。

### (a) ステータスファイルの片系運転適用のメリット及びデメリット

**メリット**

予備ファイルがない状況で現用ファイルの片系に障害が発生しても，処理を続行できます。このため，ステータスファイルの障害によって HiRDB が停止する可能性が低くなります。

**デメリット**

片系運転中に正常な系に障害が発生したり，又はステータスファイルの更新中に HiRDB が異常終了したりすると，現用ファイルの内容が失われるため，HiRDB を再開始できなくなります。

### (b) 指定方法

ユニット用ステータスファイルの片系運転をする場合は，ユニット制御情報定義で pd_syssts_singleoperation = continue を指定してください。サーバ用ステータスファイルの片系運転をする場合は，サーバ定義で pd_sts_singleoperation = continue を指定してください。なお，pd_syssts_singleoperation と pd_sts_singleoperation の指定値を同じにしてください。

**● ほかのオペランドとの関連**

pd_syssts_singleoperation 及び pd_syssts_initial_error オペランド，又は pd_sts_singleoperation 及び pd_sts_initial_error オペランドの指定値の組み合わせによって，HiRDB の起動時にステータスファイルの障害を検知した場合の HiRDB の動作が決定します。したがって，これら二つのオペランドの指定値は一緒に考えるようにしてください。HiRDB の起動時にステータスファイルの障害を検知した場合の HiRDB の動作については，マニュアル「HiRDB Version 8 システム定義」の，pd_syssts_initial_error 又は pd_sts_initial_error オペランドの説明を参照してください。

### (c) 適用の目安

ステータスファイルの片系運転の適用の目安を次に示します。

- HiRDB が再開始できない状態を避けることを重視した運用の場合は，適用しないでください。
- HiRDB がオンラインダウンとなる状態を避けることを重視した運用の場合は，適用してください。
- 系切り替え構成を適用している場合など，HiRDB の再開始を自動で行う運用の場合は，適用しないでください。

### (d) 片系運転適用時の注意

片系運転適用の有無による HiRDB の動作及び HiRDB 管理者の処置について次の表に示します。ステータスファイルに障害が発生したときの対処方法については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

表 8-3 片系運転適用の有無による HiRDB の動作及び HiRDB 管理者の処置

| 条件 | | ステータスファイルの片系運転（pd_syssts_singleoperation 又は pd_sts_singleoperation オペランドの指定値） | |
|---|---|---|---|
| | | 適用する（continue 指定） | 適用しない（省略又は stop 指定） |
| 予備ファイルがある | 現用ファイルに障害が発生 | HiRDB の動作<br>ステータスファイルをスワップします。<br>HiRDB 管理者の処置<br>障害が発生したステータスファイルの障害対策をしてください。 | |
| | 現用ファイルの両系に，同時に障害が発生 | HiRDB の動作<br>異常終了します。HiRDB は再開始できません。<br>HiRDB 管理者の処置<br>マニュアル「HiRDB Version 8 システム運用ガイド」の「ステータスファイルに障害が発生したときの対処方法」を参照してください。 | |
| 予備ファイルがない | 現用ファイルの片系に障害が発生 | HiRDB の動作<br>片系運転を適用し，処理を続行します。<br>HiRDB 管理者の処置<br>早急に予備のファイルを作成して，両系運転の状態に戻してください。 | HiRDB の動作<br>異常終了します。<br>HiRDB 管理者の処置<br>予備ファイルを作成し，HiRDB を再開始してください。 |
| | 現用ファイルの両系に，同時に障害が発生 | HiRDB の動作<br>異常終了します。HiRDB は再開始できません。<br>HiRDB 管理者の処置<br>マニュアル「HiRDB Version 8 システム運用ガイド」の「ステータスファイルに障害が発生したときの対処方法」を参照してください。 | |
| | 片系運転中に，正常な系に障害が発生 | HiRDB の動作<br>異常終了します。HiRDB は再開始できません。<br>HiRDB 管理者の処置<br>マニュアル「HiRDB Version 8 システム運用ガイド」の「ステータスファイルに障害が発生したときの対処方法」を参照してください。 | － |

（凡例）－：該当しません。

### (5) ステータスファイルの障害に関する注意事項（重要）

- 現用ファイルの両系に同時に障害が発生した場合，HiRDB が異常終了し，再開始できなくなります。対策として，物理ディスクの多重化（ミラーリング）が考えられます。
- HiRDB の開始前に，現用ファイル（終了時の現用ファイル）を削除，又は pdstsinit コマンドでステータスファイルを初期化した場合，HiRDB は再開始できなくなります。

# 8.4 RDエリアの配置

ここでは，次に示すRDエリアを配置するときの考慮点について説明します。

- システム用RDエリア
- データディクショナリLOB用RDエリア
- ユーザ用RDエリア
- ユーザLOB用RDエリア
- リスト用RDエリア

## 8.4.1 システム用RDエリアの配置

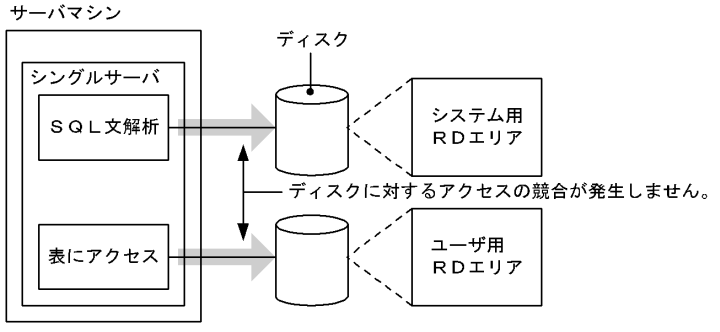
システム用RDエリアは，ユーザ用RDエリアの配置を考慮して配置します。システム用RDエリアを配置するときの考慮点を次に示します。

- **ユーザ用RDエリアを配置するディスクとは異なるディスクに配置するようにします。**

システム用RDエリアのうち，特にデータディクショナリ用RDエリアとデータディレクトリ用RDエリアは，SQL文の解析などのためにHiRDBにアクセスされることが多くなります。このため，ユーザ用RDエリアを配置するディスクと同じディスクに配置すると，SQL文の解析などのためのアクセスと，表に対するアクセスがディスク上で競合するため，どちらか一方が他方のアクセスが終了するまで待たされることになります。

ディスクアクセスの競合を発生させないためのシステム用RDエリアの配置例を次の図に示します。

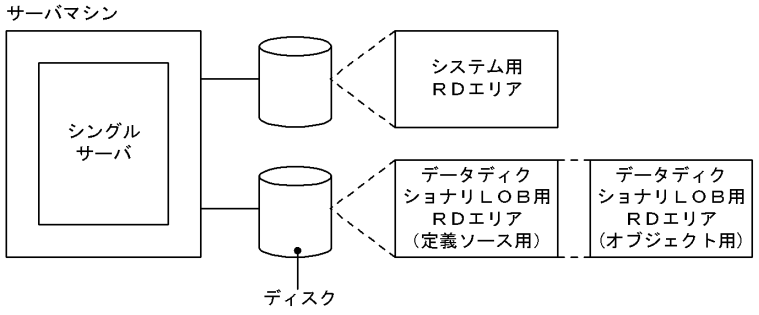
**図8-5 システム用RDエリアの配置例（HiRDB/シングルサーバの場合）**




## 8.4.2 データディクショナリLOB用RDエリアの配置

ディスクに対するアクセスの競合をなくすため，ほかのRDエリアを配置するディスクとは異なるディスクに配置するようにします。データディクショナリLOB用RDエリアの配置例を次の図に示します。

**図 8-6 データディクショナリ LOB 用 RD エリアの配置例（HiRDB/ シングルサーバの場合）**




**データディクショナリ用 RD エリアとの関連**

ストアドプロシジャ又はストアドファンクションを管理するディクショナリ表をほかのディクショナリ表とは別のデータディクショナリ用 RD エリアに格納できます。

## 8.4.3 ユーザ用 RD エリアの配置

### (1) システムログファイルとの関連

システムログファイルを配置したディスクとは異なるディスクに，ユーザ用 RD エリアを配置するようにします。このようにすることで，シンクポイント時のシステムログファイルとユーザ用 RD エリアを構成する HiRDB ファイルへの入出力処理を複数のディスクに分散できるため，シンクポイントでの処理時間を削減できます。

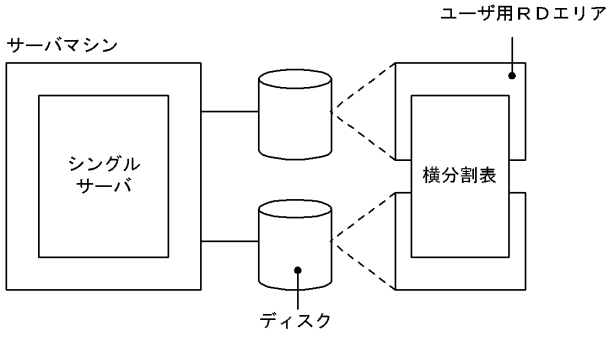
### (2) システム用 RD エリアとの関連

システム用 RD エリアを配置したディスクとは異なるディスクにユーザ用 RD エリアを配置するようにします。

### (3) 表を横分割した場合

表を横分割した場合，横分割表を格納する RD エリアを異なるディスクに配置します。ユーザ用 RD エリアの配置例を次の図に示します。

**図 8-7 ユーザ用 RD エリアの配置例（HiRDB/ シングルサーバの場合）**

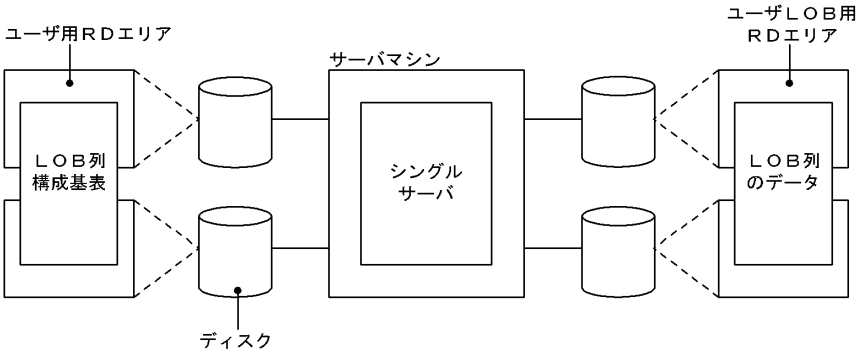


## 8.4.4 ユーザ LOB 用 RD エリアの配置

ディスクに対するアクセスの競合をなくすため，ユーザ LOB 用 RD エリア以外の RD エリアを配置したディスクとは異なるディスクにユーザ LOB 用 RD エリアを配置するようにします。

また，表を横分割した場合，横分割表を格納する RD エリアを異なるディスクに配置します。ユーザ LOB 用 RD エリアの配置例を次の図に示します。

**図 8-8 ユーザ LOB 用 RD エリアの配置例（HiRDB/シングルサーバの場合）**



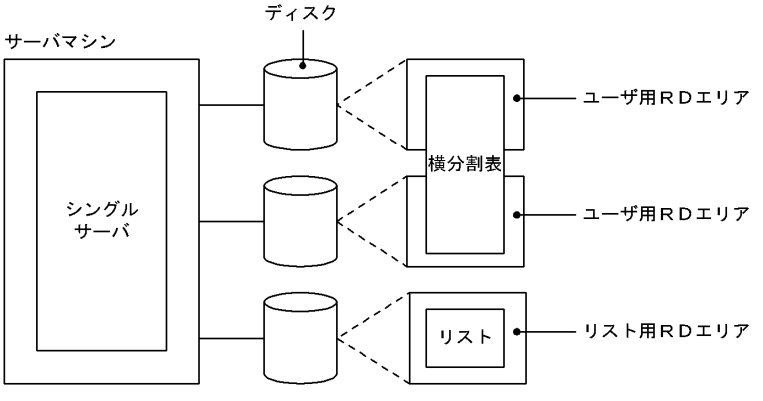

## 8.4.5 リスト用 RD エリアの配置

ディスクに対するアクセスの競合をなくすため，リスト用 RD エリア以外の RD エリアを配置したディスクとは異なるディスクにリスト用 RD エリアを配置するようにします。

なお，リスト用 RD エリアを一つ以上作成すれば，すべてのユーザ用 RD エリアに格納されている表に対するリストを作成できます。

リスト用 RD エリアの配置例を次の図に示します。

**図 8-9 リスト用 RD エリアの配置例（HiRDB/シングルサーバの場合）**



# 9 HiRDB/ パラレルサーバの設計

この章では，HiRDB/ パラレルサーバのシステム構成，HiRDB ファイルシステム領域の設計方法，システムファイルの設計方法及び RD エリアの配置方法の考慮点について説明します。


9.1 HiRDB/ パラレルサーバのシステム設計

9.2 HiRDB ファイルシステム領域の設計

9.3 システムファイルの設計

9.4 RD エリアの配置

9.5 大規模なシステムの構築

9.6 HiRDB/ パラレルサーバのヘテロ構成

# 9.1 HiRDB/ パラレルサーバのシステム設計

ここでは，HiRDB/ パラレルサーバのシステム設計とシステム構成について説明します。

なお，HiRDB External Data Access 機能を使用する場合は，併せてマニュアル「HiRDB External Data Access Version 8」を参照してください。

## 9.1.1 システム設計

### (1) サーバ構成

フロントエンドサーバ，ディクショナリサーバ及びバックエンドサーバをそれぞれ一つのサーバマシンに配置するのが，HiRDB/ パラレルサーバの基本的なサーバ構成です。ただし，各サーバマシンの CPU の処理負荷が高くない場合，一つのサーバマシンに複数のサーバを配置してもかまいません。一つのサーバマシンに複数のサーバを配置すると，必要とする共用メモリが増加します。共用メモリが不足すると，ユニットが開始できなくなるため，メモリ所要量を十分見積もってください。

設置できるサーバ数の範囲を次の表に示します。

**表 9-1 設置できるサーバ数の範囲**

| 項目 | 設置できるサーバ数の範囲 |
|---|---|
| システムマネジャ数 | 1 |
| フロントエンドサーバ数 | 1 ～ 1,024 |
| ディクショナリサーバ数 | 1 |
| バックエンドサーバ数 | 1 ～ 16,382 |
| 1 ユニットに作成できるサーバ数 | 1 ～ 34 |

### (2) システムマネジャの設置

次に示す理由のため，システムマネジャを定義するサーバマシンは HiRDB 管理者が運用しやすい場所に設置することをお勧めします。

- HiRDB 管理者はコマンドで HiRDB を操作しますが，大部分のコマンドはシステムマネジャを定義するサーバマシンから入力する必要がある

### (3) フロータブルサーバの設置

HiRDB は結合処理のような複雑な検索処理をする場合，データベースを持たないバックエンドサーバを優先的に使用して，処理性能を向上します。サーバマシンに余裕があり，複雑な検索処理をする場合にフロータブルサーバの設置を検討してください。フロータブルサーバを設置する場合，作業表用ファイル用の HiRDB ファイルシステム領域を作成してください。作業表用ファイル用の HiRDB ファイルシステム領域の名称は，バックエンドサーバ定義の pdwork オペランドで指定します。

### (4) フロントエンドサーバの複数化

SQL 処理の CPU 負荷が高く，一つのフロントエンドサーバで処理しきれない場合，フロントエンドサーバを複数設定します。これを**マルチフロントエンドサーバ**といいます。マルチフロントエンドサーバについては，「9.1.3 マルチフロントエンドサーバの設定」を参照してください。

## (5) HiRDB/パラレルサーバが使用するメモリ

HiRDB/パラレルサーバが使用するメモリについて説明します。

HiRDB/パラレルサーバは次に示すメモリを使用します。

- 共用メモリ
- プロセス固有メモリ

### (a) メモリ所要量

HiRDB/パラレルサーバが必要とするメモリ所要量をサーバマシンごとに見積もってください。HiRDB/パラレルサーバのメモリ所要量については，「15.2 HiRDB/パラレルサーバのメモリ所要量の見積もり」を参照してください。

### (b) 共用メモリのページ固定

HiRDBでは，次に示す共用メモリを実メモリ上に固定できます。

- ユニットコントローラ用共用メモリ
- グローバルバッファ用共用メモリ
- 動的変更したグローバルバッファが使用する共用メモリ
- インメモリデータバッファ用共用メモリ

共用メモリを実メモリ上に固定すると，ページの入出力が少なくなるため，性能が安定します。

#### 前提条件

共用メモリのページ固定をするための前提条件を次に示します。

##### Windowsのバージョン

Windowsでは，Windows Server 2003 Service Pack 1以降でサポートされているLarge Pageの機能を使用してページ固定をします。そのため，Large Pageがサポートされているバージョンの Windowsであることが前提となります。Windowsがページ固定に対応しているかどうかは，pdntenv -osコマンドコマンドで確認してください。

#### ページロックの権限

共用メモリのページ固定をする場合，メモリ内のページをロックできる権限が必要です。サービス実行時に使用するログオンのアカウントによって，この権限の有無が異なります。権限の設定方法を次に示します。

| サービス実行時に使用するログオンのアカウント | 権限の設定方法 |
|---|---|
| HiRDB管理者 | HiRDB管理者を，OSの［ローカル セキュリティ設定］－［ローカル ポリシー］－［ユーザー権利の割り当て］で，「メモリ内のページのロック」に設定してください。 |
| ローカルシステムアカウント | ローカルシステムアカウントにはメモリ内のページをロックできる権限があります。そのため，権限の設定は必要ありません。 |

#### 動作環境の設定

共用メモリのページ固定をするには，共用メモリの割り当て先の指定が必要です。pdntenvコマンドの-shmfileオプションでpageを指定し，共用メモリの割り当て先をページングファイル（仮想メモリ）に設定してください。

#### ページ固定の方法

共用メモリのページ固定の方法について，共用メモリの種別ごとに説明します。

- ユニットコントローラ用共用メモリ

システム共通定義，又はユニット制御情報定義の pd_shmpool_attribute オペランドに fixed を指定します。

- グローバルバッファ用共用メモリ
  システム共通定義，又はユニット制御情報定義の pd_dbbuff_attribute オペランドに fixed を指定します。
- 動的変更したグローバルバッファが使用する共用メモリ
  システム共通定義，又はユニット制御情報定義の pd_dbbuff_attribute オペランドに fixed を指定します。これによって，pdbufmod コマンドを実行して動的変更したグローバルバッファが使用する共用メモリも実メモリ上に固定されます。
- インメモリデータバッファ用共用メモリ
  pdmemdb コマンドの -p オプションに fixed を指定します。

**注意事項**

実メモリ上に連続した領域が確保できなかった場合，共用メモリのページ固定ができません。ページ固定に失敗した場合の HiRDB の動作を次に示します。

**ユニットコントローラ用，又はグローバルバッファ用共用メモリの場合**
ページ固定をしないで共用メモリを確保し処理を続行します。

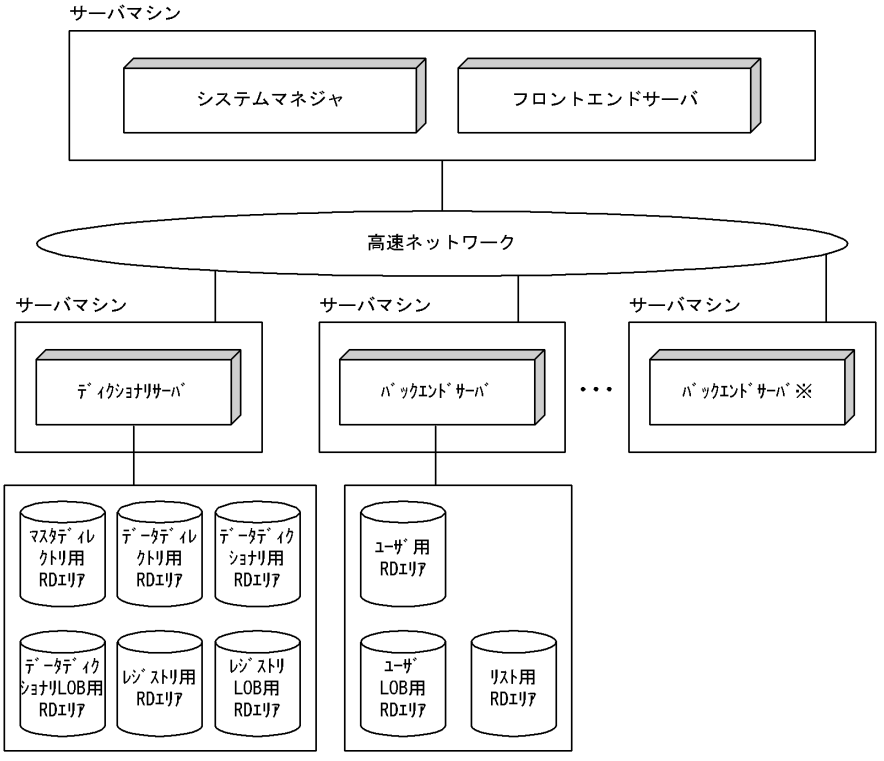
**動的変更したグローバルバッファが使用する共用メモリ，又はインメモリデータバッファ用共用メモリの場合**
HiRDB，又はコマンドが異常終了します。

## 9.1.2 HiRDB/ パラレルサーバのシステム構成

HiRDB/ パラレルサーバのシステム構成例を次の図に示します。

HiRDB/ パラレルサーバのシステム構成は，HiRDB システム定義で定義します。HiRDB システム定義の定義例については，マニュアル「HiRDB Version 8 システム定義」を参照してください。

**図** 9-1 HiRDB/ **パラレルサーバのシステム構成例**




注※ ＲＤエリアを作成しないで，フロータブルサーバとして使用します。

## 9.1.3 マルチフロントエンドサーバの設定

HiRDB/ パラレルサーバでは複数の SQL を複数のバックエンドサーバを使用して並列に処理しています。フロントエンドサーバは，SQL の解析処理，SQL の最適化処理，各バックエンドサーバへ処理の指示や検索結果の編集処理などをしています。したがって，高トラフィックなシステムではフロントエンドサーバの負荷が高くなり，そのため処理性能が向上しなくなります。こういう場合は，フロントエンドサーバを複数設定して，フロントエンドサーバの負荷を分散させてください。これを**マルチフロントエンドサーバ**といいます。

**メリット**

フロントエンドサーバが稼働するサーバマシンの処理能力ネックを解消し，スケーラブル性をより向上します。

**適用基準**

SQL 処理の CPU 負荷が高く，一つのサーバマシンだけで処理しきれない場合に適用します。

**規則**

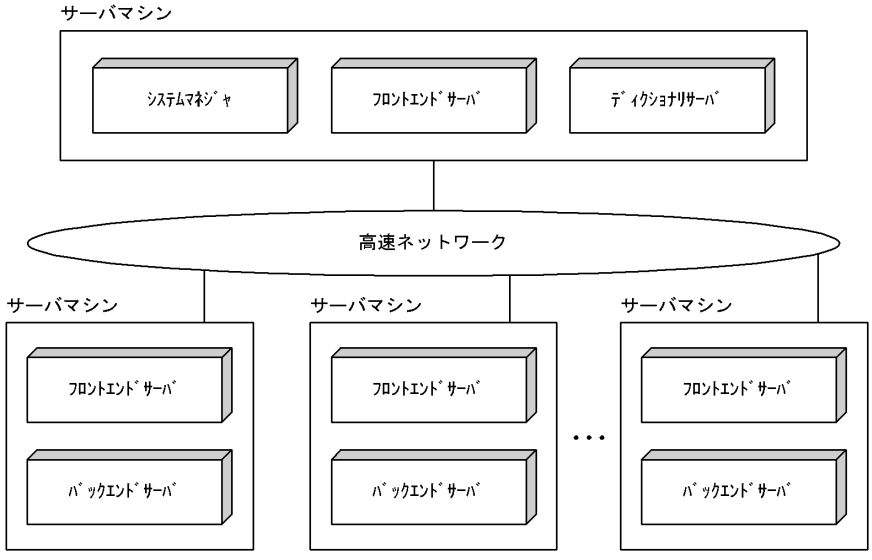
フロントエンドサーバは最大 1024 個設定できます。

**サーバマシンとの関係**

一つのユニットに複数のフロントエンドサーバを設定できません。また，特定のユニットにフロントエンドサーバを設定しないこともできます。マルチフロントエンドサーバの構成例を次の図に示しま

す。

**図 9-2 マルチフロントエンドサーバの構成例**




## (1) 接続するフロントエンドサーバの選択

複数あるフロントエンドサーバのうち，UAP がどのフロントエンドサーバに接続するかは，次に示すどちらかが決定します。

- **クライアントユーザ**
  接続するフロントエンドサーバをクライアント環境定義の PDFESHOST オペランドなどで指定します。
- HiRDB
  接続するフロントエンドサーバを HiRDB が自動的に決定します。
  接続するフロントエンドサーバをクライアント環境定義で指定しない場合，HiRDB が任意のフロントエンドサーバに UAP を接続させます。

## (2) 環境設定

マルチフロントエンドサーバを実行するために，特に指定は必要ありません。

ただし，次に示すオペランドの指定値に注意してください。このオペランドの指定値の目安については，マニュアル「HiRDB Version 8 システム定義」を参照してください。

- pd_max_dic_process
- pd_max_bes_process

## (3) HiRDB 管理者の運用

HiRDB システム定義のオペランドの指定やシステムファイルの作成個数などの環境設定方法は変わりますが，運用方法は変わりません。

### (4) マルチフロントエンドサーバ構成での，挿入又は更新時刻による並べ替え

マルチフロントエンドサーバ構成で，表定義時に CURRENT_TIMESTAMP を既定値とする DEFAULT 句を指定した時刻印型の列を含む表を，行の挿入又は更新時刻で並べ替える場合，注意が必要です。

マルチフロントエンドサーバの場合，UAP と接続したフロントエンドサーバが現在の時刻印を取得し，その値を時刻印列の既定値として設定します。ただし，フロントエンドサーバがあるユニット間でシステムの時刻が異なることがあるため，時刻印列の値で並べ替えた順番と，実際に行を挿入又は更新した時刻で並べ替えた順番が一致しない場合があります。

順番を一致させるには，表定義時に CURRENT_TIMESTAMP USING BES を既定値とする DEFAULT 句を時刻印列に指定します。USING BES を指定すると，その列に既定値を挿入又は既定値で更新時，挿入又は更新する行を格納する RD エリアを管理するバックエンドサーバで現在の時刻印を取得し，その値を挿入，又はその値で更新します。そのため，時刻印型の列で並べ替えた順番と，実際に行を挿入又は更新した時刻で並べ替えた順番は，行を格納する RD エリアを管理するバックエンドサーバがあるユニット単位で一致させることができます。

USING BES の指定有無と現在の時刻印を取得するサーバを次の表に示します。

**表 9-2 USING BES の指定有無と現在の時刻印を取得するサーバ**

| USING BES の指定有無 | 現在の時刻印を取得するサーバ | |
|---|---|---|
| | HiRDB/シングルサーバ | HiRDB/パラレルサーバ |
| なし | シングルサーバ | UAP と接続したフロントエンドサーバ |
| あり | | 挿入，又は更新する行を格納する RD エリアを管理するバックエンドサーバ |

**注意事項**

- 表を分割していて，表格納用 RD エリアを管理するバックエンドサーバが複数のユニットにわたっている場合，時刻印型の列の値で並べ替えた順番と，実際に行を挿入又は更新した時刻で並べ替えた順番とが一致しないことがあります。
- 共用表の場合，排他モードで表に排他を掛けた後でないと，時刻印型の列に既定値を挿入したり，既定値で更新したりできません。
- データベース作成ユティリティ（pdload）を使用して表にデータを格納する場合，起動したユニットでユティリティを起動した時刻印が設定されます。

### (5) クライアントユーザの運用

接続するフロントエンドサーバをクライアントユーザが決める場合は，接続するフロントエンドサーバをクライアント環境定義で指定します。**高速接続機能**を使用する場合と FES **ホストダイレクト接続機能**を使用する場合とで指定するクライアント環境定義が異なります。指定する必要があるクライアント環境定義を次の表に示します。クライアント環境定義については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

**表 9-3 マルチフロントエンドサーバ時に指定するクライアント環境定義**

| クライアント環境定義のオペランド | 接続するフロントエンドサーバを指定しない場合 | 接続するフロントエンドサーバを指定する場合 | |
|---|---|---|---|
| | | FES ホストダイレクト接続 | 高速接続 |
| PDHOST | ○ | ○ | ○ |

| クライアント環境定義のオペランド | 接続するフロントエンドサーバを指定しない場合 | 接続するフロントエンドサーバを指定する場合 | |
|---|---|---|---|
| | | FES ホストダイレクト接続 | 高速接続 |
| PDFESHOST | − | ○ | ○ |
| PDNAMEPORT | ○ | ○ | ○ |
| PDSERVICEPORT | − | − | ○ |
| PDSERVICEGRP | − | ○ | ○ |
| PDSRVTYPE | − | − | ○ |

（凡例）
○：必ず指定します。
−：指定する必要はありません。

**(a) どのフロントエンドサーバを指定するかの目安**

- アクセスする RD エリアを管理するバックエンドサーバがあるサーバマシンのフロントエンドサーバを指定することをお勧めします。
- 用途に応じて接続するフロントエンドサーバを使い分けることをお勧めします。例えば，一般の情報検索処理用，バッチ UAP 処理用，OLTP 下の UAP 処理用などにフロントエンドサーバを使い分けることをお勧めします。

**(b) HiRDB サーバへの接続時間**

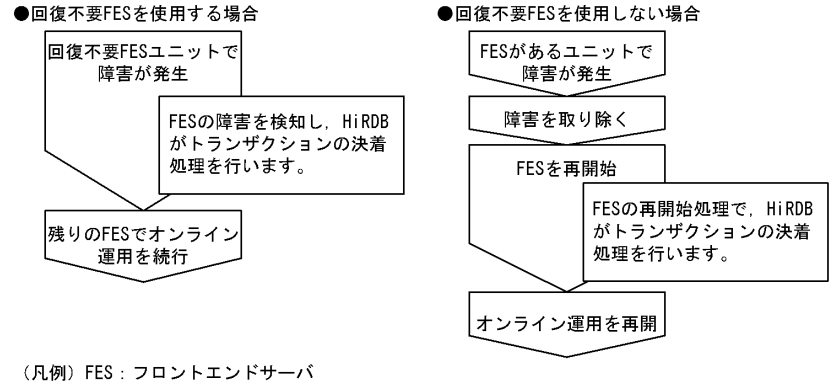
HiRDB サーバへの接続時間は，次に示す順番どおりに短縮されます（1 が最も短縮されます）。

1. 高速接続機能
2. FES ホストダイレクト接続機能
3. 接続するフロントエンドサーバを指定しない場合

## 9.1.4 回復不要 FES

フロントエンドサーバがあるユニットで障害が発生して異常終了すると，そのフロントエンドサーバから実行していたトランザクションは未決着状態になることがあります。未決着状態のトランザクションは，データベースの排他を確保しているため，一部のデータベースに対する参照又は更新が制限されます。通常，未決着状態のトランザクションの決着処理をするためには，フロントエンドサーバの障害を取り除いて再開始する必要がありますが，異常終了したフロントエンドサーバが**回復不要 FES** であれば，HiRDB が自動的に未決着状態になっていたトランザクションを決着します。これによって，ほかのフロントエンドサーバやバックエンドサーバを使用して，データベースの更新を再開できます。回復不要 FES があるユニットを**回復不要 FES ユニット**といいます。回復不要 FES を使用する場合としない場合の運用を次の図に示します。

**図 9-3 回復不要 FES を使用する場合としない場合の運用**




（凡例）FES：フロントエンドサーバ

なお，回復不要 FES を使用するためには，HiRDB Non Recover FES が必要です。

**メリット**

障害が発生したフロントエンドサーバを再開始しないで，残りのフロントエンドサーバでオンライン運用を続行できます。

**適用基準**

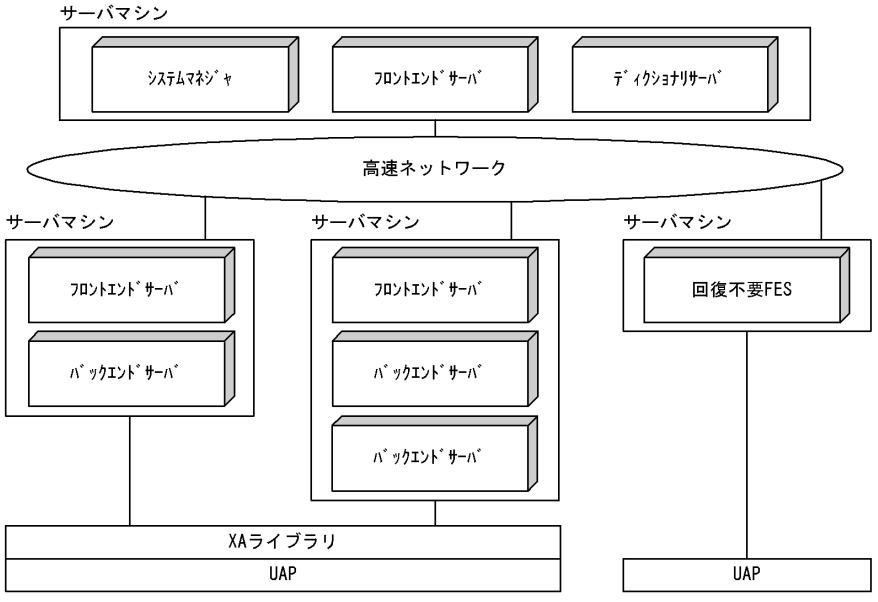
24 時間連続稼働が必要なシステムの場合に適用をお勧めします。

**ほかのフロントエンドサーバとの関係**

- 回復不要 FES は，単独で構成されるユニットに配置してください。
- 回復不要 FES では，X/Open XA インタフェースを使用して接続する UAP は実行できません。クライアント環境定義の PDFESHOST 及び PDSERVICEGRP を指定して，回復不要 FES 以外のフロントエンドサーバに接続してください。
- 回復不要 FES，及び回復不要 FES ユニットが停止していても，pdrplstart コマンド，及び pdrplstop コマンドは実行できます。

回復不要 FES を使用したシステムの構成例を次の図に示します。

**図 9-4 回復不要 FES を使用したシステムの構成例**




- 回復不要 FES では，反映側 Datareplicator の同期点処理方式に二相コミット方式を利用（反映システム定義 commitment_method オペランドに fxa_sqle を指定）した反映処理を実行できません。反映側 Datareplicator の同期点処理方式に二相コミット方式を利用する場合，反映側 HiRDB に回復不要 FES 以外のフロントエンドサーバを一つ以上配置し，反映側 Datareplicator にクライアント環境変数 PDFESHOST 及び PDSERVICEGRP を設定して，回復不要 FES 以外のフロントエンドサーバに接続してください。

**ほかの機能との関連**

- 回復不要 FES ユニットでは，系切り替え機能を適用できません。系切り替え機能を適用するシステムの場合，回復不要 FES ユニットのユニット制御情報定義の pd_ha_unit オペランドに必ず nouse を指定してください。

## (1) 設定方法

回復不要 FES を使用するには，pdstart オペランドの -k オプションに stls を指定します。

## (2) 注意事項

1. HiRDB を起動する場合に回復不要 FES ユニットが開始しないとき，HiRDB は pd_start_level オペランドの指定値に関係なく，そのユニットを除いて HiRDB を開始します。すべてのフロントエンドサーバを回復不要 FES にする場合，一つ以上のフロントエンドサーバが開始しないと HiRDB システムの起動は完了しません。
2. 回復不要 FES ユニットは独自に縮退起動をするため，pd_start_skip_unit オペランドに回復不要 FES ユニットの名称を指定しても無視します。
3. 回復不要 FES ユニットが異常終了した場合，フロントエンドサーバ及びユニットのステータス情報は STOP(A) になりますが，通常の STOP(A) とは異なり，pdstop コマンドで HiRDB のシステムマネジャとほかのユニットを正常停止，又は計画停止できます。また，回復不要 FES ユニットを強制停止した場合，フロントエンドサーバ及びユニットのステータス情報は STOP(F) になりますが，pdstop コマンドで HiRDB のシステムマネジャとほかのユニットを正常停止，又は計画停止できます。

4. 回復不要 FES ユニットは，次の場合以外，常に正常開始でユニットを開始します。
   - ユニットを正常終了以外で停止していて，かつ前回稼働時に pdstart オペランドの -k オプションに stls を指定していなかった場合
5. 回復不要 FES ユニットのステータス情報が STOP(A) になっていると，その回復不要 FES に CONNECT した UAP からの SQL 要求を HiRDB が受け付けなくなります。この場合，KFPS01820-E メッセージに表示される回復不要 FES のプロセス終了状態は「c800」になります。また，SQL で操作しようとしたデータを持つバックエンドサーバやディクショナリサーバなどのサーバプロセスの終了状態は，KFPS01820-E メッセージに「c900」と表示されることがあります。KFPS01820-E メッセージが表示された場合，プロセス終了状態が「c800」と表示されたフロントエンドサーバがあるユニットを pdstop -z で停止し，STOP(A) になった原因を対策してから再度開始してください。
6. 回復不要 FES ユニットが稼働しているのに，ネットワーク障害などで回復不要 FES ユニットのステータス情報が STOP(A) になった場合，障害が回復してシステムマネジャを配置するユニットから回復不要 FES ユニットに通信できるようになると，システムマネジャが回復不要 FES ユニットを自動的に強制停止してから再度開始します。自動的に強制停止してから再度開始する契機を次に示します。
   - システムマネジャを配置するユニットのユニット監視プロセスが，STOP(A) になっている回復不要 FES ユニットの稼動状態を確認し，KFPS05288-I メッセージを出力したとき。
   - システムマネジャを配置するユニットが再度開始する際に，システムマネジャが，STOP(A) になっている回復不要 FES ユニットの稼動状態を確認したとき。

   (1) 回復不要 FES ユニットの起動処理が正常に終了していない場合の対処

   (a) システムマネジャを配置するユニット，及びディクショナリサーバを配置するユニットの稼働状態を pdls -d ust コマンドで確認します。

   (b) これらのユニットが稼働していない場合，pdstart -q コマンドで開始します。

   (c) これらのユニットが稼働している場合，又はこれらユニットを開始しても KFPS05110-I メッセージを出力しない場合は，(2) の対処をします。

   (2) KFPS05110-I メッセージを出力しない場合の対処

   回復不要 FES ユニットの稼働状態を pdls -d ust コマンドで確認し，実行結果に応じて，次に示す対処をします。

| 回復不要 FES ユニットの稼働状態（pdls -d ust コマンドの実行結果） | 対処方法 |
|---|---|
| STOP（停止状態） | pdstart -q コマンドを実行して，回復不要 FES ユニットを再開始します。 |
| PAUSE（プロセスサーバプロセスの再起動中断状態） | 1. KFPS00715-E メッセージ，及びそれ以前にイベントログに出力されたメッセージを参照して，障害要因を取り除いた後，HiRDB のサービスを再起動します。<br>2. pdstart -q コマンドを実行して，回復不要 FES ユニットを再開始します。 |
| STARTING（開始途中） | 1. pdstop -z コマンドを実行して，回復不要 FES ユニットを強制終了します。<br>2. pdstart -q コマンドを実行して，回復不要 FES ユニットを再開始します。 |
| ONLINE（稼働状態） | |
| STOPPING（停止途中） | |

   なお，システムマネジャから回復不要 FES ユニットに通信できるようになる前にユニットが停止した場合は，システムマネジャは強制停止及び再度開始をしません。
7. 回復不要 FES から分岐して，ほかのサーバで実行しているトランザクションは，コミット決着時に分

岐先のサーバ間で決着の同期合わせを行います。このため，同期合わせのときに，分岐先サーバのいずれかがトランザクション処理を実行できない状態（系切り替え中，サーバ停止状態，サーバ開始準備中，又はサーバ停止準備中）になっていると，トランザクション第 1 状態が READY 又は COMMIT で待ち合わせを行うことがあります。この場合，トランザクション処理を実行できない状態になっている原因を該当するサーバで対策して，トランザクションの決着処理が続行できるようにしてください。

8. 回復不要 FES 機能を使用する FES に接続して実行したトランザクションは，トランザクション第 1 状態，第 2 状態に関係なく，pdcmt，pdrbk，pdfgt コマンドを使用してトランザクションを強制的に終了できないことがあります。この場合，マニュアル「HiRDB Version 8 システム運用ガイド」の「未決着状態のトランザクションを決着する方法」を参照してトランザクションを自動決着させてください。

# 9.2 HiRDB ファイルシステム領域の設計

HiRDB のシステム構築時に，HiRDB ファイルを作成する HiRDB ファイルシステム領域を作成します。ここでは，HiRDB ファイルシステム領域を作成するときの設計方針について説明します。

HiRDB ファイルシステム領域は，次に示す用途ごとに作成することをお勧めします。これによって，用途やアクセス特性の異なるファイルへの入出力が競合することを回避できます。

- RD エリア用
- 共用 RD エリア用
- システムファイル用
- 作業表用ファイル用
- ユティリティ用
- リスト用 RD エリア用

## 9.2.1 RD エリア用の HiRDB ファイルシステム領域の設計

RD エリア用の HiRDB ファイルシステム領域の設計方針について説明します。

### (1) 信頼性向上のための方針

1. RD エリア用の HiRDB ファイルシステム領域は，次に示すサーバを定義するサーバマシンに作成します。
   - ディクショナリサーバ
   - バックエンドサーバ
2. 次に示す RD エリアを作成する HiRDB ファイルシステム領域は，ディクショナリサーバを定義するサーバマシンに作成します。
   - システム用 RD エリア
   - データディクショナリ LOB 用 RD エリア
   - レジストリ用 RD エリア
   - レジストリ LOB 用 RD エリア
3. 次に示す RD エリアを作成する HiRDB ファイルシステム領域は，バックエンドサーバを定義するサーバマシンに作成します。
   - ユーザ用 RD エリア
   - ユーザ LOB 用 RD エリア

### (2) 性能向上のための方針

1. RD エリアを作成する HiRDB ファイルシステム領域は，次に示す RD エリア用ごとに作成することをお勧めします。
   - システム用 RD エリア
   - データディクショナリ LOB 用 RD エリア
   - ユーザ用 RD エリア
   - ユーザ LOB 用 RD エリア
   - レジストリ用 RD エリア
   - レジストリ LOB 用 RD エリア
2. システムファイル用の HiRDB ファイルシステム領域と，RD エリア用の HiRDB ファイルシステム領域は，別々のハードディスクに作成することをお勧めします。これによって，シンクポイントダンプを

取得するときに入出力の分散ができ，シンクポイントダンプの取得処理時間を短縮できます。

## 9.2.2 システムファイル用の HiRDB ファイルシステム領域の設計

システムファイル用の HiRDB ファイルシステム領域の設計方針について説明します。

### (1) 信頼性向上のための方針

1. システムファイル用の HiRDB ファイルシステム領域は二つ以上作成してください。一つしか作成しないと，システムファイルがあるハードディスクに障害が発生した場合，HiRDB が稼働できなくなります。
2. システムファイル用の HiRDB ファイルシステム領域は別々のハードディスクに作成してください。そうすれば，どちらかのハードディスクに障害が発生しても，HiRDB を再開始できます。

### (2) 性能向上のための方針

システムファイル用の HiRDB ファイルシステム領域と，RD エリア用の HiRDB ファイルシステム領域は，別々のハードディスクに作成することをお勧めします。これによって，シンクポイントダンプを取得するときに入出力の分散ができ，シンクポイントダンプの取得処理時間を短縮できます。

## 9.2.3 作業表用ファイル用の HiRDB ファイルシステム領域の設計

作業表用ファイル用の HiRDB ファイルシステム領域の設計方針について説明します。

### (1) 設計方針

1. 作業表用ファイル用の HiRDB ファイルシステム領域長は，その HiRDB ファイルシステム領域に作成する作業表用ファイルの総容量より大きくしてください。なお，pdfmkfs コマンドで -a オプションを指定すると，HiRDB ファイルシステム領域を自動的に拡張できます。作業表用ファイルの総容量が HiRDB ファイルシステム領域長に達した場合，自動で HiRDB ファイルシステム領域を拡張できるため，-a オプションを指定することをお勧めします。※
   作業表用ファイルの容量については，「18. 作業表用ファイルの容量の見積もり」を参照してください。
2. 作業表用ファイル用の HiRDB ファイルシステム領域は，次に示すサーバを定義するサーバマシンに作成してください。
   - ディクショナリサーバ
   - バックエンドサーバ

注※

HiRDB 再開始時に作業表用ファイル用の HiRDB ファイルシステム領域のディスク占有サイズを削減したい場合は，HiRDB 再開始前に pdfmkfs コマンドを実行して作業表用ファイル用の HiRDB ファイルシステム領域を再初期化してください。

### (2) 最大使用量を調べる方法

作業表用ファイル用の HiRDB ファイルシステム領域の最大使用量を次に示す方法で調べられます。

pdfstatfs -d **作業表用の** HiRDB **ファイルシステム領域名**

-d：

HiRDB ファイルシステム領域の最大使用量を表示するオプションです。出力情報の peak capacity が最大使用量です。なお，上記の最大使用量は pdfstatfs コマンドでクリアできます。

pdfstatfs -c **作業表用の** HiRDB **ファイルシステム領域名**

-c：

HiRDB ファイルシステム領域の最大使用量をクリアするオプションです。

## 9.2.4 ユティリティ用の HiRDB ファイルシステム領域の設計

ユティリティ用（バックアップファイル，アンロードデータファイル，又はアンロードログファイル作成用）の HiRDB ファイルシステム領域の設計方針について説明します。ユティリティ用の HiRDB ファイルシステム領域には，次に示すファイルを作成します。

- バックアップファイル
- アンロードデータファイル
- アンロードログファイル
- 差分バックアップ管理ファイル

### (1) 設計方針

1. バックアップファイル作成用にする場合は，HiRDB ファイルシステム領域長をバックアップ対象 RD エリアの総容量より大きくしてください。RD エリアの容量については，「16. RD エリアの容量の見積もり」を参照してください。
2. 差分バックアップ管理ファイル作成用の HiRDB ファイルシステム領域は，システムマネジャがあるサーバマシンに作成してください。
3. アンロードログファイル作成用にする場合は，pdfmkfs コマンドのオプションに次に示す値を指定してください。
   - -k オプション：使用目的には UTL（ユティリティ用の HiRDB ファイルシステム領域）を指定します。
   - -n オプション：HiRDB ファイルシステム領域長には次に示す計算式の値を指定します。

     **（アンロードするシステムログファイルの総レコード数**※1**×システムログファイルのレコード長）**※2**×作成するアンロードログファイル数×** 1.2 ÷ 1048576
   - -l オプション：最大ファイル数には，作成するアンロードログファイル数を指定します。
   - -e オプション：最大増分回数には，作成するアンロードログファイル数× 24 を指定します。

注※ 1

システムログファイルの自動拡張機能を適用している場合，pd_log_auto_expand_size オペランドの拡張上限サイズに指定した値で計算してください。

注※ 2

システムログファイルの概算値です。

### (2) 最大使用量を調べる方法

ユティリティ用の HiRDB ファイルシステム領域の最大使用量を次に示す方法で調べられます。

pdfstatfs -d **ユティリティ用の** HiRDB **ファイルシステム領域名**

-d：

HiRDB ファイルシステム領域の最大使用量を表示するオプションです。出力情報の peak capacity が最大使用量です。なお，上記の最大使用量は pdfstatfs コマンドでクリアできます。

pdfstatfs -c **ユティリティ用の** HiRDB **ファイルシステム領域名**

-c：

HiRDB ファイルシステム領域の最大使用量をクリアするオプションです。

## 9.2.5 リスト用 RD エリア用の HiRDB ファイルシステム領域の設計

リスト用 RD エリア用の HiRDB ファイルシステム領域の設計方針について説明します。

### (1) 設計方針

1. リストは検索の一時的な中間結果を保存するものなので，ほかの RD エリアほど信頼性は要求されません。
2. リスト用 RD エリア用の HiRDB ファイルシステム領域は，基表と同じバックエンドサーバに作成してください。

### (2) 性能向上のための方針

1. リスト用 RD エリア用の HiRDB ファイルシステム領域は，次に示す HiRDB ファイルシステム領域とは別々のハードディスクに作成することをお勧めします。別々のハードディスクに作成すると，リストを検索するときに入出力を分散できるため，処理時間を短縮できます。
   - ユーザ用 RD エリア用の HiRDB ファイルシステム領域
   - ユーザ LOB 用 RD エリア用の HiRDB ファイルシステム領域
   - 作業表用ファイル用の HiRDB ファイルシステム領域

## 9.2.6 HiRDB ファイルシステム領域の最大長

HiRDB ファイルシステム領域の最大長を次の表に示します。

**表 9-4 HiRDB ファイルシステム領域の最大長**

| 条件 | HiRDB ファイルシステム領域の最大長 |
|---|---|
| pd_large_file_use ＝ N 指定時 | 2,047 メガバイト |
| pd_large_file_use=Y 指定時（省略値） | 1,048,575 メガバイト |

**注意事項**

HiRDB ファイルシステム領域長が 100 ギガバイトを超える場合は，pd_ntfs_cache_disable=Y を指定してください。pd_ntfs_cache_disable=Y を指定する場合は，LOB 用 RD エリアとそのほかの RD エリアを同じ HiRDB ファイルシステム領域に格納しないようにしてください。

## 9.2.7 系切り替え機能使用時の注意事項

系切り替え機能で使用する HiRDB ファイルシステム領域を作成する場合の注意事項を次に示します。

- pdfmkfs コマンドの -i オプションを指定してください。-i オプションを指定しないと，HiRDB ファイルシステム領域の拡張とサーバマシンの電源異常が同時に発生した場合にファイルが破壊されることがあります。
- pdfmkfs コマンドの -k オプションに SVR（省略値）を指定しないでください。-k オプションに SVR を指定して作成した HiRDB ファイルシステム領域に対して，HiRDB は Windows のファイルキャッシュに書き込んだ時点で処理を続行します。このため，ディスク上にデータが書き込まれる前に異常が発生するとファイルの整合性がとれなくなります。

# 9.3 システムファイルの設計

ここでは，システムファイルの設計方針について説明します。

## 9.3.1 システムログファイルの設計

システムログファイルのスワップ時に，スワップ先にできる状態のシステムログファイルがないとHiRDB（ユニット）は異常終了します。これを予防するため，HiRDB にはシステムログファイルの空き容量を監視する機能（**システムログファイルの空き容量監視機能**）があります。この機能は，システムログファイルの空き率が**警告値**未満になったときに作動します。次に示す二つのレベルのどちらかを選択できます。

**レベル 1：**

システムログファイルの空き率が警告値未満になった場合，警告メッセージ KFPS01162-W を出力します。

**レベル 2：**

システムログファイルの空き率が警告値未満になった場合，新規トランザクションのスケジューリングを抑止して，サーバ内の全トランザクションを強制終了します。このとき，KFPS01160-E メッセージを出力します。これによって，システムログの出力量を抑えます。

レベル 2 を選択した場合，システムログファイルの空き容量が不足したときにサーバ内の全トランザクションが強制終了されます。このため，システムログファイルの設計をより正確に行う必要があります。

システムログファイルの設計方針について説明します。

### (1) 設計方針

1. システムログファイルはシステムマネジャを除いた各サーバに必要です。
2. サーバ内の全システムログファイルのレコード長及びレコード数を同じにしてください。
3. 各サーバに作成できるシステムログファイルは，2 ～ 200 グループです（ただし，6 グループ以上作成することを推奨します）。
   システムログファイルの容量不足によって異常終了した HiRDB を再開始する場合，作成済みのシステムログファイルと同じ数のシステムログファイルを追加する必要があります。例えば，最大量（100 ギガバイト）のシステムログファイルを 50 個で 50 グループ作成している場合は，更に最大量のシステムログファイルを 50 個で 50 グループ追加します。したがって，作成するシステムログファイルは 100 グループ以内とすることをお勧めします。
4. 1 サーバに作成できるシステムログファイルの総容量の上限は 20 テラバイトになります。
5. アンロードする回数を少なくしたい場合は，システムログファイルの 1 ファイル容量を大きくします。
6. HiRDB 運用ディレクトリがあるディスクに大量に入出力が発生するファイル（システムログのアンロードログファイルなど）を作成しないでください。
7. 一つのシステムログファイルの容量は，次に示す条件を満たす必要があります。

一つのシステムログファイルの容量（バイト）≧↑（a ＋ 368）÷ c ↑× c × b × d

a：pd_log_max_data_size オペランドの値

b：pd_log_sdinterval オペランドの値

c：pd_log_rec_leng オペランドの値

d：pd_spd_assurance_count オペランドの値

8. 全システムログファイルの容量（二重化している場合は片系の容量だけを対象とする）は，次に示す二つの条件を満たす必要があります。

**条件 1**

「17.1.1(1) システムログファイルの総容量の求め方」で見積もった容量以上の値にしてください。

**条件 2**

長大トランザクションが終了するまでは，長大トランザクション開始以降のシステムログファイルの上書きができません。また，シンクポイントダンプファイルの有効保証世代とカレント世代のシステムログファイルは上書きできません。そのため，これらの分のシステムログファイル容量を確保してください。次の計算式で求めます。

全システムログファイルの総容量（バイト）≧ 3 × a ×（b ＋ 1）

a：

データベースを更新するトランザクションのうち，実行時間が最大のトランザクション実行中に該当するサーバで出力されることのあるシステムログ量
システムログ量の見積もり式については，「17.1　システムログファイルの容量の見積もり」を参照してください。

**b**：pd_spd_assurance_count オペランドの値
シンクポイントダンプファイルの有効保証世代数です。

### (a) システムログファイルの世代数と運用への影響

システムログファイルの総容量が同じ場合，システムログファイルの世代数によって，1 世代当たりの容量が変化します。システムログファイルの世代数と運用への影響を次の表に示します。システムログファイルの総容量は同じとします。

**表 9-5　システムログファイルの世代数と運用への影響**

| 比較項目 | システムログファイルの構成 | |
|---|---|---|
| | 世代数を少なくした場合 | 世代数を多くした場合 |
| システムログファイル 1 世代当たりの容量 | 大きくなります。 | 小さくなります。 |
| スワップ間隔 | システムログファイル 1 世代当たりの容量が大きくなるため，スワップ間隔は長くなります。 | システムログファイル 1 世代当たりの容量が小さくなるため，スワップ間隔は短くなります。 |
| アンロード回数 | スワップ間隔が長くなるため，アンロード回数は少なくなります。 | スワップ間隔が短くなるため，アンロード回数は多くなります。 |
| ディスク障害などでシステムログファイルの数世代が使用できなくなった場合のシステムログ容量への影響 | • システムログファイル 1 世代当たりの容量が大きくなるため，ディスク障害時にデータベースの回復に用いるログ量は多くなり，データベース回復に掛かる時間は長くなります。<br>• システムログ容量の減少幅が大きいため，システムログの容量が減少したことによる HiRDB の稼働に及ぼす影響は大きくなります。 | • システムログファイル 1 世代当たりの容量が小さくなるため，ディスク障害時にデータベースの回復に用いるログ量は少なくなり，データベース回復に掛かる時間は短くなります。<br>• システムログ容量の減少幅が小さいため，システムログの容量が減少したことによる HiRDB の稼働に及ぼす影響は小さくなります。 |

通常の運用では，システムログファイルの世代数を少なくした場合の方がスワップ間隔及びアンロード回数の点で利点があります。ただし，障害発生時には，世代数を多くした場合の方が障害の影響が小さくな

ります。

## (2) 信頼性向上のための方針

### (a) システムログファイルの二重化

システムログファイルを二重化すると，HiRDB は両方の系に同じシステムログを取得します。取得したシステムログを読み込むとき，片方のファイルに異常が発生しても，もう一方のファイルからシステムログを読み込めるため，システムの信頼性を向上できます。なお，二重化する場合，ディスクをミラー化して二重化するより，HiRDB 管理下で二重化することをお勧めします。システムログファイルを二重化する場合は，それぞれの系のファイルを別々のハードディスクに作成してください。

システムログファイルを二重化する場合は，サーバ定義で次に示すオペランドを指定してください。

- pd_log_dual = Y
- pdlogadpf オペランドの -b オプション（B 系のシステムログファイル名を指定します）

### (b) システムログファイルの片系運転

システムログファイルの片系運転は，システムログファイルを二重化する場合に適用されます。

システムログファイルに障害が発生して，両系とも使用できるシステムログファイルがない場合でも，HiRDB のユニットを異常終了しないで正常な系だけで処理を続行できます。これを**システムログファイルの片系運転**といいます。システムログファイルの片系運転をする場合は，サーバ定義で pd_log_singleoperation = Y を指定してください。

システムログファイルの片系運転に対し，両方のシステムログファイルで処理を続行すること（通常の処理形態）を**システムログファイルの両系運転**といいます。

### (c) システムログファイルの自動オープン

HiRDB を再開始するときに，上書きできる状態のシステムログファイルがない場合，予約のファイルがあれば HiRDB が予約のファイルをオープンして上書きできる状態にし，処理を続行します。これを**システムログファイルの自動オープン**といいます。

システムログファイルの自動オープンをする場合は，サーバ定義で pd_log_rerun_reserved_file_open = Y を指定してください。

## (3) システムログファイルのレコード長

システムログファイルのレコード長を pdloginit コマンドの -l オプションで指定します。レコード長は 1024，2048，4096 のどれかを選択できます。

### (a) 新規に HiRDB を構築する場合

新規に HiRDB を構築する場合は，レコード長を 1024 にすることをお勧めします。この場合，サーバ定義の pd_log_rec_leng オペランドで 1024 を指定してください。

### (b) 既に HiRDB を稼働している場合

レコード長が 1024 以外の場合，1024 に変更することをお勧めします。

システムログファイルのレコード長の変更方法については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

**レコード長を変更するときの考慮点**

- 全システムログファイルの総容量が変更になります。全システムログファイルの総容量の求め方に

ついては，「17.1.1　システムログファイルの総容量」を参照してください。

- ユニットコントローラが使用する共用メモリサイズが変更になります。ユニットコントローラが使用する共用メモリサイズの求め方については，「15.2.3　ユニットコントローラが使用する共用メモリの計算式」を参照してください。

### (4) システムログファイルの定義

作成したシステムログファイルをどのファイルグループに対応させるかをサーバ定義の pdlogadfg 及び pdlogadpf オペランドで定義します。

## 9.3.2 シンクポイントダンプファイルの設計

シンクポイントダンプファイルの設計方針について説明します。

### (1) 設計方針

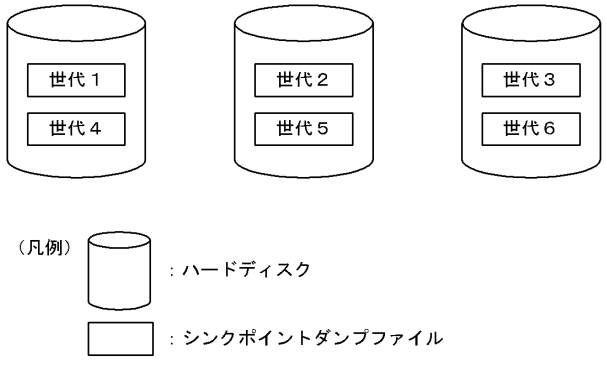
1. シンクポイントダンプファイルはシステムマネジャを除いた各サーバに必要です。
2. 各サーバに作成できるシンクポイントダンプファイルは，2 ～ 60 グループです。pdlogadfg -d spd オペランドに ONL を指定した場合は，2 ～ 30 グループです。
3. HiRDB は pdlogadfg -d spd オペランドの指定順にシンクポイントダンプファイルを使用します。
4. 各サーバに 4 個以上のシンクポイントダンプファイルを作成することをお勧めします。
5. シンクポイントダンプファイルの容量が不足すると，HiRDB を開始できません。このため，シンクポイントダンプファイルのレコード数は，システム共通定義の最大同時接続数（pd_max_users オペランド）の指定値より大きな値にしてください。詳細なシンクポイントダンプファイルの容量計算式については，「17.2　シンクポイントダンプファイルの容量の見積もり」を参照してください。

### (2) 信頼性向上のための方針

#### (a) ファイル構成例

ハードディスク障害に備えて，シンクポイントダンプファイルをそれぞれ別々のハードディスクに作成してください。そのような構成を取れない場合は，隣り合わせの世代を別々のハードディスクに作成するような構成にしてください。例を次の図に示します。

**図 9-5　隣り合わせの世代を別々のハードディスクに作成する構成例**




#### (b) シンクポイントダンプファイルの二重化

シンクポイントダンプファイルを二重化すると，HiRDB は A 系及び B 系の両方に同じシンクポイントダンプを取得します。取得したシンクポイントダンプを読み込むとき，片方のファイルに異常が発生しても，もう一方のファイルからシンクポイントダンプを読み込めるため，システムの信頼性を向上できます。

また，二重化している場合，有効保証世代数を 1 世代にすると，信頼性を損ねることなく上書きできない状態のシンクポイントダンプファイル数を削減できます。

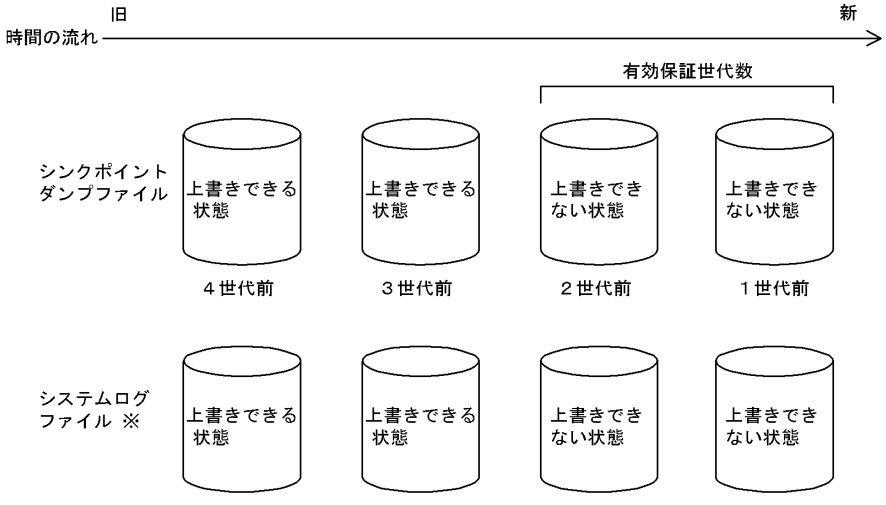
シンクポイントダンプファイルを二重化する場合は，サーバ定義で次に示すオペランドを指定してください。

- pd_spd_dual = Y
- pdlogadpf オペランドの -b オプション（B 系のシステムログファイル名を指定します）

### (c) シンクポイントダンプファイルの有効保証世代数

一つのシンクポイントダンプファイルには，HiRDB が 1 回に取得するシンクポイントダンプが格納されます。HiRDB は，シンクポイントダンプファイルを世代という概念で管理しています。HiRDB 管理者は，何世代前までのシンクポイントダンプファイルに対応するシステムログファイルを上書きできない状態にするかを指定できます。これを**シンクポイントダンプファイルの有効保証世代数**といいます。シンクポイントダンプファイルの有効保証世代数を次の図に示します。

**図 9-6 シンクポイントダンプファイルの有効保証世代数**




注※ シンクポイントダンプファイルに対応するシステムログファイルを示します。

〔説明〕

有効保証世代を 2 とすると，2 世代前までのシンクポイントダンプファイル及びそのシンクポイントダンプファイルに対応するシステムログファイルが，上書きできない状態になります。3 世代前より前の世代のシンクポイントダンプファイル及びそのシンクポイントダンプファイルに対応するシステムログファイルは，上書きできる状態になります。

シンクポイントダンプファイルの運用に必要なファイル数は，**有効保証世代数＋ 1** となります。シンクポイントダンプファイルの有効保証世代数は，サーバ定義の pd_spd_assurance_count オペランドで指定してください。

なお，シンクポイントダンプファイルを二重化している場合，必要な有効保証世代数は 1 世代をお勧めします。二重化していない場合，2 世代をお勧めします。

#### (d) シンクポイントダンプファイルの縮退運転

シンクポイントダンプファイルに障害が発生して，運用に必要なファイル数（有効保証世代数＋ 1）を下回った場合でも，最低二つのファイルで処理を続行できます。これを**シンクポイントダンプファイルの縮退運転**といいます。

シンクポイントダンプファイルの縮退運転をする場合は，サーバ定義の pd_spd_reduced_mode オペランドを指定してください。

#### (e) シンクポイントダンプファイルの自動オープン

シンクポイントダンプファイルに障害が発生して，運用に必要なファイル数（有効保証世代数＋ 1）を下回った場合，予約のファイルがあれば HiRDB が予約のファイルをオープンして上書きできる状態にし，処理を続行します。これを**シンクポイントダンプファイルの自動オープン**といいます。

シンクポイントダンプの自動オープンをする場合は，サーバ定義で pd_spd_reserved_file_auto_open = Y を指定してください。

### (3) シンクポイントダンプファイルの定義

作成したシンクポイントダンプファイルをどのファイルグループに対応させるかを pdlogadfg 及び pdlogadpf オペランドで定義します。

なお，pdlogadfg オペランドだけを指定しておくと，HiRDB 稼働中にシンクポイントダンプファイルを追加できます。

## 9.3.3 ステータスファイルの設計

ステータスファイルの設計方針について説明します。

### (1) 設計方針

1. 両系のファイルに障害が起きないように，A 系と B 系のファイルは別々のディスクに作成します。
2. ステータスファイルの容量不足による HiRDB の異常終了を防ぐため，見積もったファイル容量よりも大きい容量の予備ファイルを幾つか作成してください。ステータスファイルは，容量が満杯になると予備ファイルとスワップしますが，満杯になったステータスファイルと予備ファイルの容量が同じ場合，スワップ先のファイルでも容量不足になり，HiRDB は異常終了します。そのため，例えばステータスファイルを 6 組作成する場合，そのうちの 2 組以上のファイル容量をほかのファイル容量より大きくすることをお勧めします。
3. 各サーバマシンにユニット用ステータスファイルが必要です。
4. システムマネジャを除いた各サーバにサーバ用ステータスファイルが必要です。
5. A 系と B 系のファイルのレコード長及びレコード数を同じにしてください。
6. 一つのユニットに作成できるユニット用ステータスファイルは 1 ～ 7 組です。
7. 一つのサーバに作成できるサーバ用ステータスファイルは 1 ～ 7 組です。

### (2) 信頼性向上のための方針

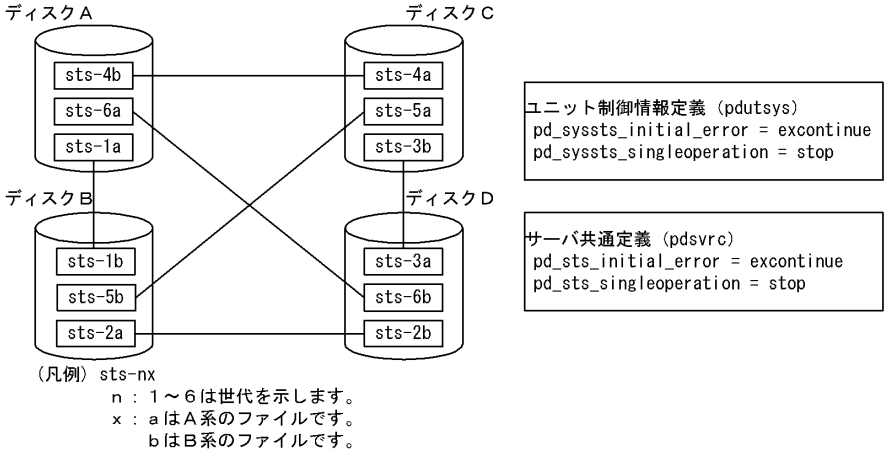
1. ステータスファイルは 3 組（二重化× 3=6 ファイル）以上用意し，ディスク障害によってすべてのステータスファイルが障害とならないように配置します。
2. 容量不足による HiRDB の異常終了を防ぐため，ステータスファイルの容量は見積もった容量の 1.2 倍以上の容量を準備することをお勧めします。
3. ステータスファイルには，HiRDB の再開始処理でシステムの状態を回復するために必要な情報を格納しています。予備ファイルがない状態で現用ファイルに障害が発生すると，システムの状態が回復できません。したがって，常に予備ファイルがあるように運用し，現用ファイルの障害に備えてください。

### (a) お勧めする構成

ディスク障害発生時から回復時までの安全性を考えると，ステータスファイルは四つのディスクに 6 組（二重化× 6=12 ファイル）用意し，次の図のように配置することをお勧めします。また，片系運転中に正常な系に障害が発生すると，HiRDB を再開始できなくなるため，ステータスファイルの片系運転は適用しない（pd_syssts_singleoperation 及び pd_sts_singleoperation に stop を指定）ことをお勧めします。

四つのディスクにステータスファイルを 6 組配置する例を次の図に示します。

**図 9-7 四つのディスクに 6 組のステータスファイルを配置する例**




〔説明〕

このように配置すると，あるディスクで障害が発生した後に，更に別のディスクで障害が発生しても，残りの二つのディスクに両系とも正常なファイルが残るため，障害が発生していないディスクのステータスファイルを現用として HiRDB を稼働し続けることができます。例えば，ディスク A に障害が発生し，その後ディスク B にも障害が発生した場合でも，ディスク C 及び D にある両系のステータスファイル（sts-3a と sts-3b）を現用ファイルとして稼働し続けます。この状態で，更に現用ファイルの片系に障害が発生した場合，HiRDB は異常終了しますが，現用ファイルの片系ファイルが正常のため，障害が発生したディスクのどれか一つを回復すると HiRDB を再開始できます。

## (3) ステータスファイルの定義

pdstsinit コマンドで作成したステータスファイルをどの論理ファイルに対応させるかを pd_syssts_file_name_1 ～ 7 及び pd_sts_file_name_1 ～ 7 オペランドで定義します。

ユニット用ステータスファイルは，pd_syssts_file_name_1 ～ 7 オペランドで定義します。サーバ用ステータスファイルは，pd_sts_file_name_1 ～ 7 オペランドで定義します。

なお，pd_syssts_file_name_2 ～ 7 オペランド又は pd_sts_file_name_2 ～ 7 オペランドに，実体のないステータスファイルを定義しておくと，HiRDB 稼働中にステータスファイルを追加できます。ただし，この場合，次に示すオペランドを指定しておく必要があります。

**ユニット用ステータスファイルの場合**

- pd_syssts_initial_error
- pd_syssts_last_active_file

**サーバ用ステータスファイルの場合**

- pd_sts_initial_error
- pd_sts_last_active_file

## (4) ステータスファイルの片系運転

予備ファイルがない状況で現用ファイルの片系に障害が発生した場合，正常な系（片方の系）だけで処理を続行することを**ステータスファイルの片系運転**といいます。ステータスファイルが片系運転になると，KFPS01044-I メッセージが出力されます。

片系運転中に現用ファイルに障害が発生すると，HiRDB を再開始できなくなるため，ステータスファイルの片系運転の適用は推奨しません。ステータスファイルの組数を増やし，予備ファイルがない状況が発生しないような運用をしてください。

なお，ステータスファイルの片系運転に対し，両方の系のステータスファイルで処理を続行すること（通常の処理形態）を**ステータスファイルの両系運転**といいます。

### (a) ステータスファイルの片系運転適用のメリット及びデメリット

**メリット**

予備ファイルがない状況で現用ファイルの片系に障害が発生しても，処理を続行できます。このため，ステータスファイルの障害によって HiRDB が停止する可能性が低くなります。

**デメリット**

片系運転中に正常な系に障害が発生したり，又はステータスファイルの更新中に HiRDB が異常終了したりすると，現用ファイルの内容が失われるため，HiRDB を再開始できなくなります。

### (b) 指定方法

ユニット用ステータスファイルの片系運転をする場合は，ユニット制御情報定義で pd_syssts_singleoperation = continue を指定してください。サーバ用ステータスファイルの片系運転をする場合は，サーバ定義で pd_sts_singleoperation = continue を指定してください。なお，pd_syssts_singleoperation と pd_sts_singleoperation の指定値を同じにしてください。

**● ほかのオペランドとの関連**

pd_syssts_singleoperation 及び pd_syssts_initial_error オペランド，又は pd_sts_singleoperation 及び pd_sts_initial_error オペランドの指定値の組み合わせによって，HiRDB の起動時にステータスファイルの障害を検知した場合の HiRDB の動作が決定します。したがって，これら二つのオペランドの指定値は一緒に考えるようにしてください。HiRDB の起動時にステータスファイルの障害を検知した場合の HiRDB の動作については，マニュアル「HiRDB Version 8 システム定義」の，pd_syssts_initial_error 又は pd_sts_initial_error オペランドの説明を参照してください。

### (c) 適用の目安

ステータスファイルの片系運転の適用の目安を次に示します。

- HiRDB が再開始できない状態を避けることを重視した運用の場合は，適用しないでください。
- HiRDB がオンラインダウンとなる状態を避けることを重視した運用の場合は，適用してください。
- 系切り替え構成を適用している場合など，HiRDB の再開始を自動で行う運用の場合は，適用しないでください。

### (d) 片系運転適用時の注意

片系運転適用の有無による HiRDB の動作及び HiRDB 管理者の処置について次の表に示します。ステータスファイルに障害が発生したときの対処方法については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

表 9-6 片系運転適用の有無による HiRDB の動作及び HiRDB 管理者の処置

| 条件 | | ステータスファイルの片系運転（pd_syssts_singleoperation 又は pd_sts_singleoperation オペランドの指定値） | |
|---|---|---|---|
| | | 適用する（continue 指定） | 適用しない（省略又は stop 指定） |
| 予備ファイルがある | 現用ファイルに障害が発生 | HiRDB の動作<br>ステータスファイルをスワップします。<br>HiRDB 管理者の処置<br>障害が発生したステータスファイルの障害対策をしてください。 | |
| | 現用ファイルの両系に，同時に障害が発生 | HiRDB の動作<br>異常終了します。HiRDB は再開始できません。<br>HiRDB 管理者の処置<br>マニュアル「HiRDB Version 8 システム運用ガイド」の「ステータスファイルに障害が発生したときの対処方法」を参照してください。 | |
| 予備ファイルがない | 現用ファイルの片系に障害が発生 | HiRDB の動作<br>片系運転を適用し，処理を続行します。<br>HiRDB 管理者の処置<br>早急に予備のファイルを作成して，両系運転の状態に戻してください。 | HiRDB の動作<br>異常終了します。<br>HiRDB 管理者の処置<br>予備ファイルを作成し，HiRDB を再開始してください。 |
| | 現用ファイルの両系に，同時に障害が発生 | HiRDB の動作<br>異常終了します。HiRDB は再開始できません。<br>HiRDB 管理者の処置<br>マニュアル「HiRDB Version 8 システム運用ガイド」の「ステータスファイルに障害が発生したときの対処方法」を参照してください。 | |
| | 片系運転中に，正常な系に障害が発生 | HiRDB の動作<br>異常終了します。HiRDB は再開始できません。<br>HiRDB 管理者の処置<br>マニュアル「HiRDB Version 8 システム運用ガイド」の「ステータスファイルに障害が発生したときの対処方法」を参照してください。 | － |

（凡例）－：該当しません。

### (5) ステータスファイルの障害に関する注意事項（重要）

- 現用ファイルの両系に同時に障害が発生した場合，HiRDB が異常終了し，再開始できなくなります。対策として，物理ディスクの多重化（ミラーリング）が考えられます。
- HiRDB の開始前に，現用ファイル（終了時の現用ファイル）を削除，又は pdstsinit コマンドでステータスファイルを初期化した場合，HiRDB は再開始できなくなります。

# 9.4 RD エリアの配置

ここでは，次に示す RD エリアを配置するときの考慮点について説明します。

- システム用 RD エリア
- データディクショナリ LOB 用 RD エリア
- ユーザ用 RD エリア
- ユーザ LOB 用 RD エリア
- リスト用 RD エリア

## 9.4.1 システム用 RD エリアの配置

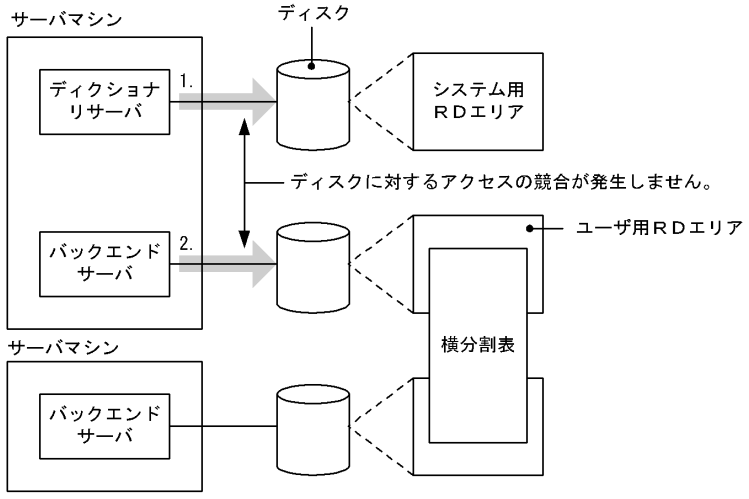
システム用 RD エリアは，ユーザ用 RD エリアの配置を考慮して配置します。システム用 RD エリアを配置するときの考慮点を次に示します。

- **システム用 RD エリアはディクショナリサーバに配置します。**
- **ディクショナリサーバとバックエンドサーバが同一のサーバマシンにある場合は，ユーザ用 RD エリアを配置するディスクとは異なるディスクにシステム用 RD エリアを配置するようにします。**

システム用 RD エリアのうち，特にデータディクショナリ用 RD エリアとデータディレクトリ用 RD エリアは，SQL 文の解析などのために HiRDB にアクセスされることが多くなります。このため，ユーザ用 RD エリアを配置するディスクと同じディスクに配置すると，SQL 文の解析などのためのアクセスと，表に対するアクセスがディスク上で競合するため，どちらか一方が他方のアクセスが終了するまで待たされることになります。

ディスクアクセスの競合を発生させないためのシステム用 RD エリアの配置例を次の図に示します。

**図 9-8 システム用 RD エリアの配置例（HiRDB/ パラレルサーバの場合）**




（凡例）

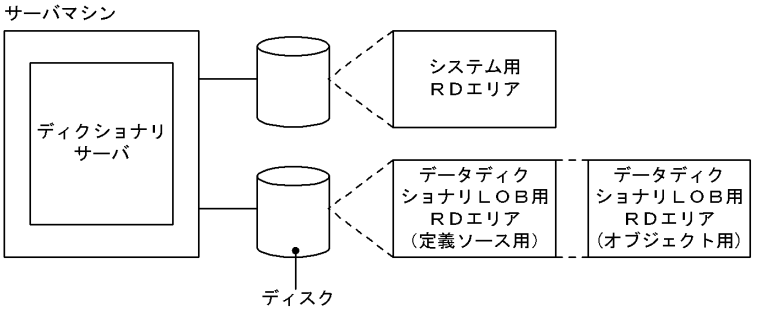
：ディスクに対するアクセス

1. ＳＱＬ文の解析などのためのアクセス
2. 表に対するアクセス

## 9.4.2 データディクショナリ LOB 用 RD エリアの配置

ディスクに対するアクセスの競合をなくすため，ほかの RD エリアを配置するディスクとは異なるディスクに配置するようにします。データディクショナリ LOB 用 RD エリアの配置例を次の図に示します。

**図 9-9 データディクショナリ LOB 用 RD エリアの配置例（HiRDB/パラレルサーバの場合）**




**データディクショナリ用 RD エリアとの関連**

ストアドプロシジャ又はストアドファンクションを管理するディクショナリ表をほかのディクショナリ表とは別のデータディクショナリ用 RD エリアに格納できます。

## 9.4.3 ユーザ用 RD エリアの配置

### (1) システムログファイルとの関連

システムログファイルを配置したディスクとは異なるディスクに，ユーザ用 RD エリアを配置するようにします。このようにすることで，シンクポイント時のシステムログファイルとユーザ用 RD エリアを構成する HiRDB ファイルへの入出力処理を複数のディスクに分散できるため，シンクポイントでの処理時間を削減できます。

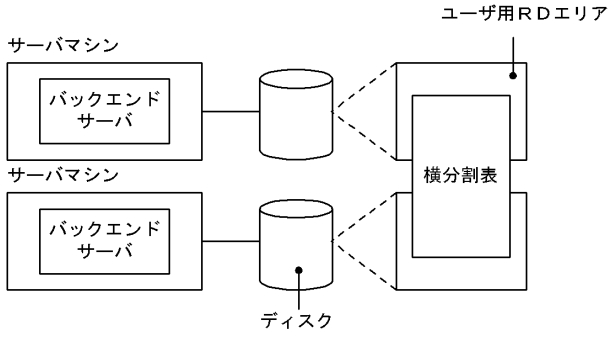
### (2) システム用 RD エリアとの関連

システム用 RD エリアを配置したディスクとは異なるディスクにユーザ用 RD エリアを配置するようにします。

### (3) 表を横分割した場合

表を横分割した場合，横分割表を格納する RD エリアを異なるバックエンドサーバに配置します。また，横分割表を格納する RD エリアを異なるディスクに配置します。ユーザ用 RD エリアの配置例を次の図に示します。

**図 9-10 ユーザ用 RD エリアの配置例（HiRDB/パラレルサーバの場合）**




### (4) フロータブルサーバの設置

複数のバックエンドサーバにわたる結合処理やソート処理などの，表に対する複雑な問い合わせ処理をするときは，ユーザ用 RD エリアの配置に注意します。

すべてのバックエンドサーバにユーザ用 RD エリアを配置すると，あるバックエンドサーバがユーザ用 RD エリアに対するアクセスのほか，複雑な問い合わせ処理をするため，このバックエンドサーバの負荷が高くなります。このため，システム全体のスループットが低下することがあります。

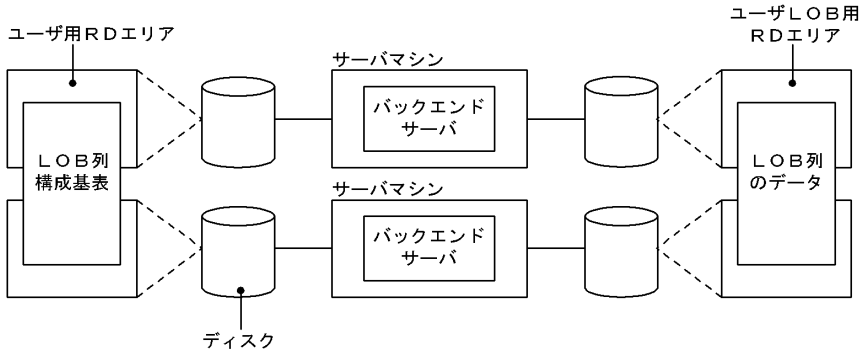
よって，サーバマシンの台数に余裕がある場合，ユーザ用 RD エリアを配置しないバックエンドサーバ（フロータブルサーバ）を設置します。フロータブルサーバを設置すると，複雑な問い合わせ処理をフロータブルサーバに割り当てるため，各バックエンドサーバの負荷を軽減できます。

## 9.4.4 ユーザ LOB 用 RD エリアの配置

ディスクに対するアクセスの競合をなくすため，ユーザ LOB 用 RD エリア以外の RD エリアを配置したディスクとは異なるディスクにユーザ LOB 用 RD エリアを配置するようにします。

また，LOB データを格納しているユーザ LOB 用 RD エリアと，LOB 列構成基表を格納しているユーザ用 RD エリアとは同一のバックエンドサーバに配置します。ユーザ LOB 用 RD エリアの配置例を次の図に示します。

**図 9-11 ユーザ LOB 用 RD エリアの配置例（HiRDB/パラレルサーバの場合）**

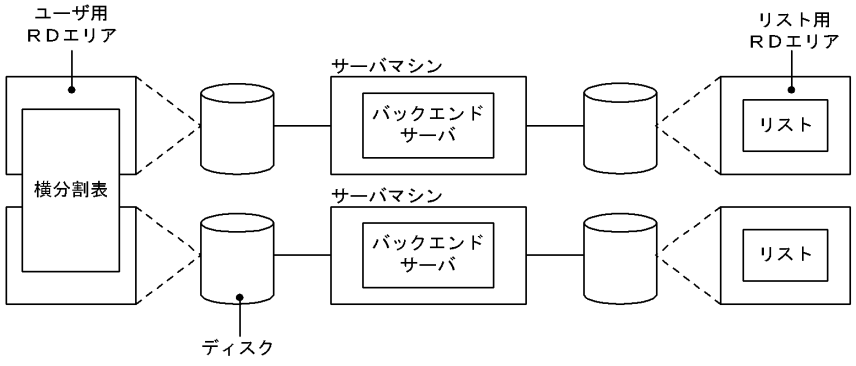


## 9.4.5 リスト用 RD エリアの配置

基表があるバックエンドサーバにリスト用 RD エリアを配置します。リスト用 RD エリアを一つ以上作成すれば，そのバックエンドサーバにあるすべての表に対するリストを作成できます。

また，ディスクに対するアクセスの競合をなくすため，リスト用 RD エリア以外の RD エリアを配置したディスクとは異なるディスクにリスト用 RD エリアを配置するようにします。リスト用 RD エリアの配置例を次の図に示します。

**図 9-12 リスト用 RD エリアの配置例（HiRDB/パラレルサーバの場合）**



# 9.5 大規模なシステムの構築

HiRDB/ パラレルサーバで 64 ユニットを超える大規模なシステムを構築する場合は，次の点を考慮してください。

## (1) 通信資源

HiRDB で使用するポートを固定することによって，通信負荷を減らせます。HiRDB で使用するポートを固定する場合には，システム定義で次のように指定してください。

- pd_name_fixed_port_lookup オペランドに Y を指定して，自ユニットの共用メモリ情報で他ユニットとの通信をするようにしてください。また，併せて次のオペランドを指定してください。
  - pd_mlg_port オペランド，又は pdunit オペランドの -m オプション
  - pd_alv_port オペランド，又は pdunit オペランドの -a オプション
  - pd_trn_port オペランド，又は pdunit オペランドの -t オプション
  - pd_scd_port オペランド，又は pdunit オペランドの -s オプション
  - pd_name_port オペランド，又は pdunit オペランドの -p オプション

また，通信処理の性能向上のため，次の指定をしてください。

- pd_ipc_conn_nblock_time オペランドに 2 を指定してください。HiRDB のサーバ間通信の場合に，稼働していないサーバがあったとき，このオペランドの指定値が大き過ぎるとタイムアウトまでの待ち時間が延びてしまいます。
- pd_bes_connection_hold オペランドに Y を指定してください。
- UAP の接続時間（SQL の CONNECT から DISCONNECT までの間）が短いシステムの場合は，pd_bes_conn_hold_trn_interval オペランドに 0 を指定してください。
- クライアント側では，高速接続機能を使用してください。高速接続機能については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

## (2) 注意事項

- トランザクションが複数のバックエンドサーバにわたるデータをアクセスする場合，サーバ間のデータ通信経路を極力減らすようにしてください。次のどれかに該当する場合，複数のバックエンドサーバにわたるデータをアクセスします。
  - 複数のバックエンドサーバに横分割した表を指定した SQL を発行する
  - FROM 句に二つ以上の表を指定した SQL を発行する
  - 副問合せを指定した SQL を発行する
  - 集合演算を指定した SQL を発行する
  - 共用表のデータを更新する SQL を発行する
  - 同一トランザクション内に複数の SQL を発行する場合，それぞれの SQL の FROM 句に異なるバックエンドサーバに定義された表を指定する

  上記のどれかに該当する場合は，次の対策をすることでサーバ間のデータ通信経路を削減できます。
  - 横分割表の分割数を少なくする
  - SQL の探索条件に分割キーに対する条件を指定する
  - 表の結合時は，対象となる表の分割を合わせ，結合キーを分割キーとする
  - トランザクションがアクセスする複数の表のデータを同じバックエンドサーバに格納する
- フロータブルサーバを極力使用しないでください。次のどれかに該当する場合，フロータブルサーバを使用します。
  - FROM 句に二つ以上の表を指定した SQL を発行する（結合方式にネストループジョインを適用する場合を除く）

- 副問合せを指定した SQL を発行する
- 集合演算を指定した SQL を発行する
- ORDER BY 句を指定した SQL を発行する（ORDER BY のためのソート処理をしなくても，インデクスを検索することで ORDER BY に含まれる列のソート順を保証できる場合を除く）
- GROUP BY 句を指定した SQL を発行する
- DISTINCT を指定した SQL を発行する
- ビュー表，WITH 句，又は FROM 句に導出表を指定した SQL を発行する（ビュー表，WITH 句を指定した SQL で内部導出表を作成しない場合を除く）
  内部導出表の条件については，マニュアル「HiRDB Version 8 SQL リファレンス」を参照してください。
- FOR READ ONLY 句を指定した SQL を発行する

上記に該当する場合は，次の対策をすることで使用するフロータブルサーバ数を削減できることがあります。

- SQL 最適化オプションに "FLTS_ONLY_DATA_BES" を指定する
- SQL 最適化オプションに "SORT_DATA_BES" を指定する

SQL 最適化オプションについては，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

- 64 ユニットを超える構成の場合には，MIB パフォーマンス情報監視機能は使用できません。
- 複数のフロントエンドサーバを経由して共用表の更新をしないでください。共用表の更新をする場合には，クライアントが接続するフロントエンドサーバができるだけ同じになるようにしてください。

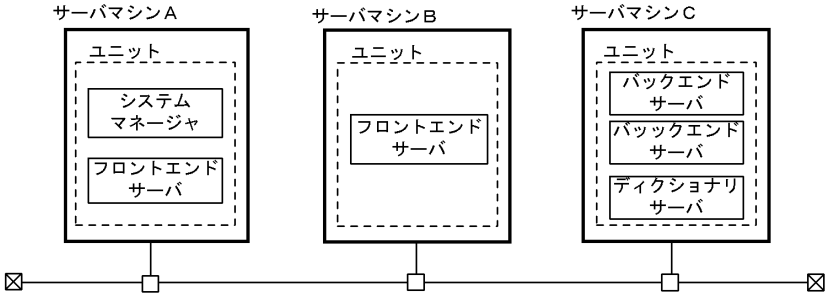
# 9.6 HiRDB/ パラレルサーバのヘテロ構成

通常，HiRDB/ パラレルサーバのすべてのユニットは同じプラットフォームである必要がありますが，次の条件を満たす場合，異なるプラットフォームが混在したヘテロ構成の HiRDB/ パラレルサーバが構築できます。

- すべてのフロントエンドサーバは同じプラットフォーム（Windows Server 2003 (IPF) 又は Windows (x64)）上で動作する
- すべてのバックエンドサーバ及びディクショナリサーバは同じプラットフォーム（Windows Server 2003 (IPF) 又は Windows (x64)）上で動作する
- システムマネジャは，フロントエンドサーバ又はバックエンドサーバと同じプラットフォーム上で動作する

ヘテロ構成の例を次の図に示します。

**図 9-13 HiRDB/ パラレルサーバのヘテロ構成の例**




〔説明〕

- この HiRDB/ パラレルサーバは 3 台のサーバマシンで構成され，サーバマシン A 及び B はアプリケーションサーバ，C はデータベースサーバとして使用する例です。
- サーバマシン A 及び B のプラットフォームは Windows (x64)，サーバマシン C のプラットフォームは Windows Server 2003 (IPF) です。

**注意事項**

ヘテロ構成を適用している場合にバックエンドサーバ又はバックエンドサーバがあるユニットを移動するときは，次に示す規則を守ってください。

- バックエンドサーバ又はバックエンドサーバがあるユニットを移動する場合，移動前後のサーバマシンのプラットフォームを同じにしてください。
  異なるプラットフォーム上への移動はしないでください。

# 10 マルチ HiRDB の設計

この章では，マルチ HiRDB のシステム設計で検討する項目について説明します。

10.1 マルチ HiRDB のシステム設計

10.2 運用上の注意

# 10.1　マルチ HiRDB のシステム設計

マルチ HiRDB のシステムを設計する場合に，通常の HiRDB と設計方法が異なる箇所についてだけ説明します。

## 10.1.1　マルチ HiRDB のインストール

マルチ HiRDB をインストールする場合に気を付けることについて説明します。

### (1) HiRDB 管理者の登録

マルチ HiRDB のサーバでは，HiRDB ごとに別々の HiRDB 管理者を登録してください。HiRDB 管理者の登録については，「2.1.2　HiRDB 管理者の登録」を参照してください。

### (2) HiRDB をインストールするときの注意

1. 既に HiRDB が「標準セットアップ」でインストールされているサーバマシンに，更に HiRDB をインストールする場合は，インストール時に「識別子付きセットアップ」を選んで，**セットアップ識別子**を指定してください。セットアップ識別子には，単一サーバマシンで一意な 4 文字以内の空白を含まない任意の半角英数字（英文字は大文字）を指定します。
2. セットアップ識別子は HiRDB の環境を識別するために指定します。
3. HiRDB ごとに異なるインストールディレクトリを指定してインストールしてください。
4. 複数の HiRDB をインストールできるのは HiRDB 05-05 以降です。05-04 以前のバージョンと混在しないでください。
5. HiRDB/ シングルサーバと HiRDB/ パラレルサーバを混在できません。
6. マルチ HiRDB に対応していない関連製品を使用する場合，HiRDB を標準インストール（セットアップ識別子の付かない形式）を選択してインストールする必要があります。

HiRDB のインストールについては，「2.2　HiRDB のインストール手順」を参照してください。

### (3) 連絡用ポートのサービス名及びサービスポート番号の設定（HiRDB/ パラレルサーバの場合又は HiRDB/ シングルサーバで系切り替え機能を使用する場合）

セットアップ識別子を指定してインストールした場合，HiRDB のサービスが起動する前に連絡用ポートのサービス名及び連絡用ポートのサービス番号を SERVICES ファイルに設定する必要があります。連絡用ポートのサービス名とサービスポート番号の決定から設定まで手順を次に示します。

#### (a) システムの構成の確認

HiRDB システムで同一，単一サーバマシンで一意な連絡用ポートのサービス名及びサービスポート番号を決定します。HiRDB のユニットをどのように構成するかを確認してください。

#### (b) サービスポート番号の OS への登録

(a) で決定した連絡用ポートのサービス名と，HiRDB システムで同一，単一サーバマシンで一意なサービスポート番号を SERVICES ファイル（「%windir%¥system32¥drivers¥etc」下のファイル）に登録します。

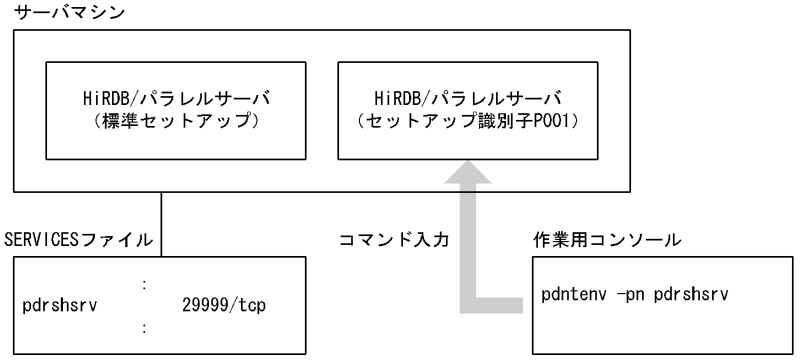
#### (c) 連絡用ポートのサービス名の変更

(b) で設定した連絡用ポートのサービス名を各ユニットに設定します。HiRDB サーバ内のユニットは同じ連絡用ポートのサービス名，サービスポート番号にしてください。サービスポート番号は，pdntenv コ

マンドで設定します。

インストール直後の連絡用ポートのサービス名には，標準セットアップした場合は pdrshsrv に，セットアップ識別子を指定してインストールした場合は pdrshsrv に，セットアップ識別子を付けた名称が仮定されます。標準セットアップした HiRDB/ パラレルサーバとセットアップ識別子（P001）を指定した HiRDB/ パラレルサーバで，マルチ HiRDB を構成したときの連絡用ポートのサービス名とサービスポート番号の設定例を次の図に示します。

**図 10-1　連絡用ポートのサービス名とサービスポート番号の設定例**




〔説明〕

1. SERVICES ファイルに連絡用ポートのサービス名 pdrshsrv とサービスポート番号 29999 を設定します。
2. セットアップ識別子 P001 の HiRDB/ パラレルサーバの連絡用ポートのサービス名を pdntenv コマンドで pdrshsrv に変更します。

## 10.1.2　マルチ HiRDB の環境設定

### (1) HiRDB サーバの環境変数の設定

マルチ HiRDB のサーバで別々に設定した HiRDB 管理者は，それぞれが操作する対象の HiRDB を環境変数 PDDIR で判別します。HiRDB 管理者別に，HiRDB 運用ディレクトリを環境変数 PDDIR に指定します。

また，環境変数 PATH に各 HiRDB の %PDDIR%¥bin を設定すると，PATH 中で先に指定した HiRDB の運用コマンドしか使用できません。そこで，各 HiRDB を操作し分けるには，それぞれの HiRDB 用のウィンドウを用意し，それぞれのウィンドウで環境変数を設定して運用することをお勧めします。

### (2) HiRDB システム定義の設定

HiRDB ごとに HiRDB システム定義を作成してください。HiRDB システム定義に指定する次に示す情報は，HiRDB ごとに変えてください。

- HiRDB 識別子（システム共通定義の pd_system_id オペランド）
- HiRDB のポート番号（システム共通定義の pd_name_port オペランド）
- ユニット識別子（ユニット制御情報定義の pd_unit_id オペランド）

### (3) クライアント環境定義の設定

クライアントからどの HiRDB にアクセスするかは，クライアント環境定義の PDNAMEPORT オペランドで指定します。アクセスする HiRDB のポート番号を PDNAMEPORT オペランドに指定してください。クライアント環境定義については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

# 10.2 運用上の注意

## (1) 運用コマンド及びユティリティ

セットアップ識別子を指定してインストールした HiRDB に対して，HiRDB の運用コマンド及びユティリティを実行する場合は，環境変数を設定する必要があります。HiRDB では，運用コマンド及びユティリティを実行する環境に合わせた環境変数を設定した作業用コンソール及び環境変数設定用バッチファイルを提供しています。

### (a) 作業用のコンソールの使用方法

**［スタート］－［プログラム］－［HiRDBSingleServer セットアップ識別子］**又は**［HiRDBParallelServer セットアップ識別子］－［HiRDB コマンドプロンプト］**をクリックして作業用コンソールを起動します。この作業用コンソールで，運用コマンド及びユティリティを実行します。

### (b) 環境変数設定用バッチファイルの使用方法

環境変数設定用バッチファイル（%PDDIR%¥SAMPLE¥sampleconf¥HiRDBCMD.BAT）を直接カスタマイズすると，更新インストール時に上書きされるため，このバッチファイルを別名でコピーし，必要な環境変数の設定などのカスタマイズをします。そして，運用コマンド及びユティリティの実行に利用してください。

標準のコマンドプロンプト上で実行する場合は，次に示す環境変数の設定が必要です。

| 環境変数 | 説明 |
|---|---|
| PDDIR | HiRDB のインストールディレクトリを指定します。 |
| PDCONFPATH | HiRDB システム定義ファイルを格納するディレクトリを絶対パスで指定します。ただし，ユニット制御情報は，この指定に関係なく %PDDIR%¥conf 下のファイルを使用します。 |
| PATH | PATH 環境変数の先頭に %PDDIR%¥BIN;%PDDIR%¥CLIENT¥UTL; を追加します。 |
| PDUXPLDIR | %PDDIR%¥UXPLDIR を指定します。 |
| PDDFUXPLDIR | %PDDIR%¥DFUXPLDIR を指定します。 |

## (2) クライアントからの接続用ポート番号の設定

クライアントからの接続用の HiRDB ポート番号は，各 HiRDB で異なるポート番号を設定してください。ポート番号は，システム共通定義の pd_name_port オペランドで指定します。pd_name_port オペランドについては，マニュアル「HiRDB Version 8 システム定義」を参照してください。

## (3) セットアップ識別子付きの HiRDB の起動と終了

セットアップ識別子を指定してインストールをした場合，起動又は停止するサービス名にセットアップ識別子が付きます。サービスを起動又は停止する場合は，指定したセットアップ識別子の付いたサービス名で操作してください。

## (4) 系切り替え機能との関連

### (a) MSCS 又は MSFC への登録方法

MSCS 又は MSFC に登録して HiRDB を系切り替え構成にする場合は，現用系と予備系のセットアップ識別子を同じにしておく必要があります。

### (b) 相互系切り替え構成への移行について

MSCS 又は MSFC を利用して HiRDB を系切り替え構成にする場合は，同じサービス名を別サーバマシンで起動します。既存の環境があるときはサービス名が重複するため，既存のサービス名を変更する必要があります。サービス名を変更する場合は，既存の HiRDB をアンインストールした後，同じディレクトリに再度インストールしてください。また，ディレクトリ構成が重複していた場合は，HiRDB サーバを移行して，環境を新規に作成する必要があります。HiRDB サーバの移行については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

# 11 グローバルバッファ，ローカルバッファの設計

この章では，グローバルバッファ，ローカルバッファを設計する上で検討する項目について説明します。


11.1 **グローバルバッファの割り当て**

11.2 **グローバルバッファのバッファ面数の設定**

11.3 **プリフェッチ機能の指定**

11.4 **非同期** READ **機能の指定**

11.5 **デファードライト処理の指定**

11.6 **デファードライト処理の並列** WRITE **機能の指定**

11.7 **コミット時反映処理の設定**

11.8 **グローバルバッファの** LRU **管理方式**

11.9 **スナップショット方式によるページアクセス**

11.10 **グローバルバッファの先読み入力**

11.11 **ローカルバッファ**

# 11.1 グローバルバッファの割り当て

**グローバルバッファ**とは，ディスク上の RD エリアに格納されているデータを入出力するためのバッファの集まりのことで，共用メモリ上に確保されます。データを更新するためにバッファ上で更新され，データベースには未反映のバッファを**更新バッファ**といいます。また，データを参照するためのバッファ，及びデータベースに反映済みのバッファを**参照バッファ**といいます。

データを格納する RD エリア又はインデクスには，必ずグローバルバッファを割り当てます。グローバルバッファには次に示す種類があります。

- インデクス用グローバルバッファ
- データ用グローバルバッファ
- LOB 用グローバルバッファ

また，HiRDB の稼働中にグローバルバッファを追加，変更，又は削除することを**グローバルバッファの動的変更**といいます。動的変更をするには，pdbufmod コマンドを使用します。グローバルバッファの動的変更の詳細は，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

それぞれのグローバルバッファと割り当て方法について説明します。

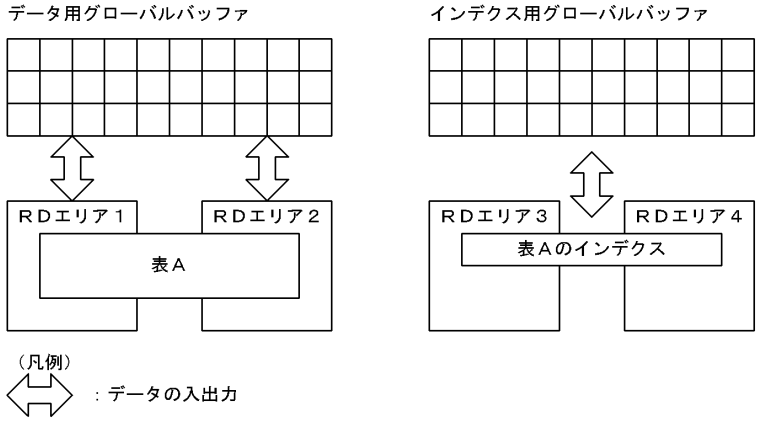
## 11.1.1 インデクス用グローバルバッファの割り当て

アクセス頻度が高いインデクスがある場合，特に，クラスタキーや UNIQUE を指定したインデクスがある場合は，そのインデクスに専用のグローバルバッファを割り当てるようにします。これでインデクスが常駐できるため，インデクスにアクセスするときの入出力回数が削減できます。

インデクス専用のグローバルバッファを割り当てると，このグローバルバッファは，表の行を格納したユーザ用 RD エリアのグローバルバッファとは独立して管理されます。このため，グローバルバッファ内でインデクスページとデータページが共用されることがありません。しかし，あるインデクスに対して，表やほかのインデクスと同一のグローバルバッファを割り当てると，ほかの表のデータなどが一時的に大量に入力された場合，このインデクスの情報がグローバルバッファ上から追い出されてしまうことがあります。

インデクス専用のグローバルバッファの概要を次の図に示します。

**図 11-1 インデクス用グローバルバッファの概要**



## 11.1.2　データ用グローバルバッファの割り当て

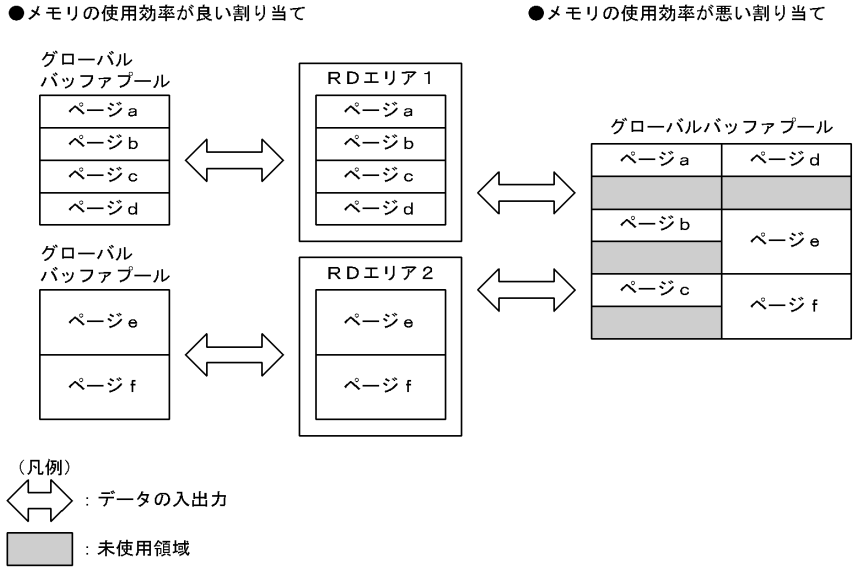
### (1) 異なるページ長の RD エリアが複数ある場合

異なるページ長の RD エリアが複数ある場合は，同じか又は近いページ長の RD エリアをまとめて一つのグローバルバッファに割り当てます。これによって，メモリの使用効率が良くなります。

異なるページ長の RD エリアをまとめて一つのグローバルバッファに割り当てると，ページ長が最も大きい RD エリアに合わせてグローバルバッファが確保されます。このため，ページ長が小さい RD エリアに対してデータページの入出力をした場合，グローバルバッファ一面に使用されない領域ができることになり，メモリの使用効率が下がります。

グローバルバッファの割り当ての例を次の図に示します。

**図 11-2　グローバルバッファの割り当ての例**




HiRDB/ パラレルサーバの場合は，サーバごとにページ長が最も大きい RD エリアに合わせてグローバルバッファが確保されます。例えば，ページ長が最も大きい RD エリアのページ長がバックエンドサーバ 1 は 4096 バイト，バックエンドサーバ 2 は 8192 バイトの場合，割り当てられるグローバルバッファサイズはバックエンドサーバ 1 が 4096 バイト，バックエンドサーバ 2 が 8192 バイトになります。

### (2) 一つのグローバルバッファに複数の RD エリアを割り当てる場合

複数の RD エリアを構成する HiRDB ファイルが一つの HiRDB ファイルシステム領域に含まれる場合は，これらの RD エリアを一つのグローバルバッファに割り当てるようにします。

### (3) UAP からのアクセス方法が異なる RD エリアが複数ある場合

同じページ長でも，UAP からのアクセス方法が異なる RD エリアが複数ある場合，例えば，用途が異なる RD エリア，順次処理が多くて更新が少ない RD エリア，追加や更新が多い RD エリアなどがある場合は，それぞれ異なるグローバルバッファを割り当てるようにします。

### （4） RD エリアの追加が予想される場合

データベース構成変更ユティリティ（pdmod）で追加できる RD エリアを次に示します。

- ユーザ用 RD エリア
- ユーザ LOB 用 RD エリア
- データディクショナリ LOB 用 RD エリア
- ストアドプロシジャを管理するディクショナリ表を格納するデータディクショナリ用 RD エリア
- リスト用 RD エリア

また，フレキシブルハッシュ分割した表を格納している RD エリアは，ALTER TABLE で RD エリアを追加できます。追加した RD エリアを使用するためには，グローバルバッファを割り当てる必要があります。このため，RD エリアの追加が予想される場合は，追加が予想される RD エリアの最大ページを見込んで，システム共通定義の pdbuffer オペランドで -o オプションを指定したグローバルバッファを用意しておきます。

なお，あらかじめグローバルバッファを割り当てていなかった場合は，システム共通定義の pdbuffer オペランドで，追加した RD エリアへの割り当てを再度定義します。これによって，追加した RD エリアを使用できます。

### （5） リスト用 RD エリアにグローバルバッファを割り当てるときの注意

リスト用 RD エリアにグローバルバッファを割り当てる場合，ユーザ用 RD エリアにグローバルバッファを割り当てるときの設計に加え，次に示す点に注意してください。

1. リスト用 RD エリアのグローバルバッファと，表やインデクスのグローバルバッファを共用したときに大量のリストを作成すると，表やインデクスのデータがグローバルバッファから追い出されてしまうことがあります。したがって，なるべく共用しないで，リスト用 RD エリア専用のグローバルバッファを割り当ててください。
2. リスト用 RD エリアのグローバルバッファのバッファ面数には，なるべく「同時にアクセスするリスト数× 1.5」以上の値を指定してください。
3. リスト用 RD エリアのグローバルバッファと，表やインデクスのグローバルバッファとを共用する場合，ページ長が同じか又はページ長が近い RD エリアを共用してください。
4. リスト用 RD エリアのグローバルバッファにプリフェッチ機能を指定した場合，次に示す SQL 文が実行されたときにセグメントサイズ分のページを一括して入力します。
   - SELECT 文でリストを介した表の検索をする場合，リストページを一括して入力します。
   - ASSIGN LIST 文でリストからリストを作成する場合，FROM 句に指定したリストのリストページを一括して入力します。

## 11.1.3 LOB 用グローバルバッファの割り当て

次に示す場合に，LOB 用 RD エリアにグローバルバッファを割り当てるようにします。これによって，LOB 用 RD エリアに格納されているデータの入出力回数が削減されます。

- プラグインインデクスを格納している場合
- データ量がそれほど多くなく，バッファリング効果が望める場合
- アクセス頻度の高い LOB データを格納している場合

なお，RD エリア間のバッファリングの干渉を避けるため，単独の RD エリアに割り当てることをお勧めします。次に示す LOB 用 RD エリアにグローバルバッファを割り当てられます。

- データディクショナリ LOB 用 RD エリア
- ユーザ LOB 用 RD エリア
- レジストリ LOB 用 RD エリア

## 11.1.4 グローバルバッファの割り当て方法

### (1) インデクス用グローバルバッファを割り当てる場合

システム共通定義の pdbuffer オペランドの -i オプションに，インデクス用グローバルバッファを割り当てるインデクスを，「認可識別子．インデクス識別子」の形式で指定します。

クラスタキーの場合はインデクス識別子を HiRDB が決定するため，クラスタキーを指定した表を定義した後に，ディクショナリ表の SQL_INDEXES 表の INDEX_NAME 列を検索してインデクス識別子を確認してください。クラスタキーのインデクス識別子は，次のように表示されます。

(CLUSTER **表番号** )

ディクショナリ表の検索方法と SQL_INDEXES 表については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

定義したインデクスにインデクス用グローバルバッファを割り当てる場合は，いったん HiRDB を正常終了し，pdbuffer オペランドを指定してインデクス用グローバルバッファを割り当ててください。この作業を行わない場合，定義したインデクスはインデクス格納 RD エリアに割り当てられているデータ用グローバルバッファを使用します。

**●グローバルバッファ面数を見積もるときの考え方**

グローバルバッファ面数は，インデクスの総ページ数（インデクスの格納ページ数として算出した値）以上とすることを原則とし，そこからインデクスの重要度によってグローバルバッファ面数を減らしてください。

インデクスの使用ページ数は，データベース状態解析ユティリティ（pddbst）で確認できます。

### (2) データ用グローバルバッファを割り当てる場合

システム共通定義の pdbuffer オペランドの -r オプションでグローバルバッファに割り当てる RD エリア名を指定します。

### (3) LOB 用グローバルバッファを割り当てる場合

次に示す手順で割り当てます。

1. システム共通定義の pdbuffer オペランドの -r オプションでグローバルバッファに割り当てる LOB 用 RD エリア名を指定します。
2. システム共通定義の pdbuffer オペランドの -b オプションでグローバルバッファに割り当てる LOB 用 RD エリア名を指定します。

### (4) グローバルバッファの定義例

**RD エリアの構成**

RD エリアの構成は次のとおりとします。

| RD エリアの種類 | RD エリア名 |
|---|---|
| マスタディレクトリ用 RD エリア | RDMAST |
| データディレクトリ用 RD エリア | RDDIR |

| RD エリアの種類 | RD エリア名 |
|---|---|
| データディクショナリ用 RD エリア | RDDIC |
| ユーザ用 RD エリア | USER01，USER02，USER03 |
| ユーザ LOB 用 RD エリア | ULOB03 |
| データディクショナリ LOB 用 RD エリア | DICLOB01，DICLOB02 |
| リスト用 RD エリア | LIST01 |

**定義例**

グローバルバッファの定義例を次に示します。

```
pdbuffer -a DGB1 -n 1000 -r RDMAST,RDDIR,RDDIC              1
pdbuffer -a DGB2 -n 1000 -r USER01,USER02
pdbuffer -a DGB3 -n 1000 -r USER03
pdbuffer -a DGB4 -n 1000 -r ULOB03
pdbuffer -a DGB5 -n 1000 -r DICLOB01
pdbuffer -a DGB6 -n 1000 -r DICLOB02
pdbuffer -a DGB7 -n 1000 -r LIST01

pdbuffer -a LGB1 -n 1000 -b ULOB03                          2
pdbuffer -a LGB2 -n 1000 -b DICLOB01
pdbuffer -a LGB3 -n 1000 -b DICLOB02

pdbuffer -a IGB1 -n 1000 -i USER1.INDX01                    3
pdbuffer -a IGB2 -n 1000 -i USER1.INDX02
```

〔説明〕

1. データ用グローバルバッファの定義です。作成するすべての RD エリアを -r オプションで指定します。
2. LOB 用グローバルバッファの定義です。
   -b オプションに指定している RD エリアは，-r オプションでも指定しておく必要があります。
3. インデクス用グローバルバッファの定義です。-i オプションにインデクスの認可識別子とインデクス識別子を指定します。

この定義例に出てくる pdbuffer オペランドのオプションを簡単に説明します。

-a：グローバルバッファの名称を指定します。

-n：グローバルバッファの面数を指定します。

-r：データ用グローバルバッファを割り当てる RD エリアを指定します。

-b：LOB 用グローバルバッファを割り当てる LOB 用 RD エリアを指定します。

-i：インデクス用グローバルバッファを割り当てるインデクスを指定します。

# 11.2 グローバルバッファのバッファ面数の設定

## (1) 共用メモリの上限を考慮した設定

グローバルバッファは，共用メモリ上に確保されます。この共用メモリは，HiRDB の運用ディレクトリがあるディスク上にファイルとして確保されます。そのため，ディスクの空き容量を十分に確保しておく必要があります。

共用メモリのセグメントサイズは，システム共通定義の SHMMAX で指定した値で最大 16 個までしか確保されません。そのため，SHMMAX 値［MB］× 16 の範囲に収まるようにグローバルバッファを定義する必要があります。

なお，1 個の共用メモリセグメントに確保できないグローバルバッファを定義すると，複数の共用メモリが確保されるため，共用メモリに対するアクセスのオーバヘッドが大きくなります。

## (2) バッファヒット率を考慮した設定

グローバルバッファは，共用メモリ上に確保されます。必要以上にグローバルバッファのバッファ面数を設定すると，共用メモリが増加して，システムのディスク容量及びメモリを圧迫します。また，グローバルバッファを検索するためのオーバヘッドも大きくなります。このため，必要最低限の入出力性能が得られるように設定する必要があります。

必要最低限の入出力性能が得られるように設定するには，グローバルバッファ全体のヒット率（更新バッファヒット率＋参照バッファヒット率）及び参照バッファヒット率が高くなるように設定します。このためには，次に示す方法があります。

- **グローバルバッファのバッファ面数を大きくします。**
- **グローバルバッファに割り当てる RD エリア又はインデクスを異なるグローバルバッファに分けます。**

なお，上記の方法でバッファ面数を設定して HiRDB を稼働してから性能向上を図る場合は，pdbufls コマンド又は統計解析ユティリティ（pdstedit）を指定します。

pdbufls コマンドの場合には，編集項目である，グローバルバッファ全体のヒット率を高くなるように面数を設定します。

統計解析ユティリティ（pdstedit）の場合には，編集項目である，更新バッファヒット率及び参照バッファヒット率を参照して，グローバルバッファ全体のヒット率が高くなるように面数を設定します。

## (3) 設定方法

システム共通定義の pdbuffer オペランドの -n オプションでバッファ面数を指定します。

# 11.3 プリフェッチ機能の指定

**プリフェッチ機能**とは，グローバルバッファ（又はローカルバッファ）上に複数のページを一括して入力することです。

## (1) プリフェッチ機能の効果

ダイレクトディスクアクセス（raw/IO 機能）を使用して大量検索をする場合に入出力時間を短縮できます。特にインデクスを使用しない検索又はインデクスを使用して昇順検索をする表で，データ件数が多い場合に有効です。

## (2) 適用基準

プリフェッチ機能は次に示す SQL 文又はユティリティの場合にページを一括して入力します。

- インデクスを使用しない SELECT，UPDATE，DELETE 文の場合に**データページ**を一括して入力します。
- インデクスを使用した昇順検索をする SELECT，UPDATE，DELETE 文（= 条件，IN 条件を除く）の場合にインデクスリーフページを一括して入力します。
- クラスタキーを使用した昇順検索をする SELECT，UPDATE，DELETE 文（= 条件，IN 条件を除く）の場合にインデクスリーフページ及びデータページを一括して入力します。
- ローカルバッファを使用しないデータベース再編成ユティリティ（pdrorg）のアンロードの場合にインデクスリーフページ及びデータページを一括して入力します。

## (3) 指定方法

### (a) グローバルバッファの場合

プリフェッチ機能を動作させるには，システム共通定義の pdbuffer オペランドの -m オプションに 1 以上を指定します。また，一括して入力するページ数は，pdbuffer オペランドの -p オプションに指定します。

### (b) ローカルバッファの場合

プリフェッチ機能を使用するには，pdlbuffer オペランドの -p オプションに一括して入力するページ数を指定します。

## (4) 指定上の考慮点

- プリフェッチ機能を使用する場合，グローバルバッファ（又はローカルバッファ）とは別に一括入力専用のバッファが取られます。これによって，グローバルバッファ用の共用メモリが増加します。グローバルバッファが使用する共用メモリの計算式については，「15. HiRDB のメモリ所要量」を参照してください。
- プリフェッチ機能が有効に動作しているかどうかは統計解析ユティリティ（pdstedit）又は pdbufls コマンドのプリフェッチヒット率を参照してください。

# 11.4 非同期 READ 機能の指定

プリフェッチ機能を使用してグローバルバッファ上に複数のページを一括入力するとき，一括入力用のバッファに DB 処理サーバプロセスから同期処理で一括入力して先読みをしています。**非同期 READ 機能**とは，プリフェッチ機能使用時に一括入力用のバッファを 2 面用意し，DB 処理が一つのバッファを使用中に DB 処理とは非同期に非同期 READ プロセスがもう一つのバッファに先読み入力をする機能です。DB 処理と先読み入力を同時に実行させることで処理時間を短縮できます。また，HiRDB/ パラレルサーバの場合は，入出力の待ち時間にスレッドを切り替えて処理することで入出力待ち時間を削減できます。

なお，非同期 READ 機能はローカルバッファには使用できません。また，SCHEDULE 属性の RD エリアには適用されません。プリフェッチ機能で動作します。

### (1) 非同期 READ 機能の効果

プリフェッチ機能と同じですが，プリフェッチ機能だけを使用した場合と比べて，非同期 READ 機能は処理負荷が高いジョイン処理などで有効です。非同期 READ 機能は，入出力処理時間が長いダイレクトディスクアクセス（raw/IO 機能）を使用している場合，特に効果があります。逆に，入出力時間が掛からない Windows のファイルシステム又は日立ディスクアレイサブシステムのディスクなどを使用している場合は余り効果が得られないことがあります。

### (2) 指定方法

プリフェッチ機能の指定（pdbuffer オペランドの -m オプションに 1 以上を指定）をしていることが前提です。

pd_max_ard_process オペランドで，非同期 READ プロセス数を指定します。0 を指定するか，又はこのオペランドを省略した場合，非同期 READ 機能は動作しません。

### (3) 指定上の考慮点

プリフェッチ機能を使用する場合，グローバルバッファとは別に一括入力専用のバッファが 2 面取られます。これによって，グローバルバッファ用の共用メモリが増加します。グローバルバッファが使用する共用メモリの計算式については，「15. HiRDB のメモリ所要量」を参照してください。

# 11.5 デファードライト処理の指定

**デファードライト処理**とは，グローバルバッファ上で更新されたページを COMMIT 文が発行されてもディスクに書き込まないで，更新ページ数がある一定の値に達した時点でディスクに書き込む処理のことです。なお，更新ページ数がある一定の値（HiRDB が決定する値）に達した時点を**デファードライトトリガ**といいます。ディスクに書き込む更新ページの数は，システム共通定義の pdbuffer オペランドの -w オプションで指定した，デファードライトトリガでの更新ページの出力比率を基に HiRDB が決定します。なお，次の RD エリアはデファードライト処理の対象ではありません。

- データディクショナリ LOB 用 RD エリア
- ユーザ LOB 用 RD エリア
- レジストリ LOB 用 RD エリア
- リスト用 RD エリア

## (1) デファードライト処理の効果

COMMIT 文が発行されてもディスクに書き込まないため，入出力処理の負荷が軽減します。

## (2) 指定方法

pd_dbsync_point オペランドに sync を指定するか，指定を省略してください。また，pdbuffer オペランドの -w オプションに，デファードライトトリガでの更新ページ出力比率を指定します。

## (3) 指定上の考慮点

1. グローバルバッファに割り当てる RD エリアに格納された表やインデクスに対する更新が多い場合は，デファードライトトリガでの更新ページの出力比率を低めに設定します。
2. グローバルバッファに対する更新が多くても，同じデータに対する更新がほとんど発生しない場合は，デファードライトトリガでの更新ページの出力比率を高めに設定します。
3. HiRDB を稼働してから，更に性能の向上を図る場合は，pdbufls コマンドを使用します。それぞれの編集項目である各グローバルバッファの更新要求ヒット率を参照して，次に示すように設定します。

- 更新要求バッファヒット率が高い場合は，デファードライトトリガでの更新ページの出力比率を低く設定します。
- 更新要求バッファヒット率が低い場合は，デファードライトトリガでの更新ページの出力比率を高く設定します。

## (4) 注意

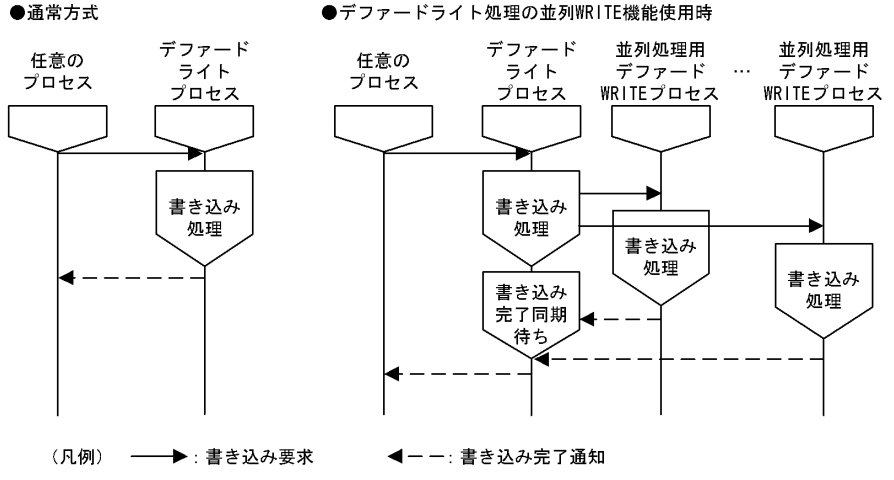
デファードライトトリガでの更新ページ出力比率を必要以上に高くすると，デファードライト処理でのディスクへの書き込みが多くなります。このため，同時に実行しているトランザクションが入出力待ちになることがあり，レスポンスタイムが悪くなることがあります。

また，必要以上に低くすると，シンクポイントダンプの出力時に，データベースに書き出すページ数が多くなることがあります。このため，同時に実行しているトランザクションが入出力待ちになることがあり，レスポンスタイムが悪くなることがあります。

# 11.6　デファードライト処理の並列 WRITE 機能の指定

**デファードライト処理の並列** WRITE **機能**とは，デファードライトの書き込み処理を複数のプロセスで並列に実行する機能です。並列 WRITE 機能を使用する場合としない場合の概要を次の図に示します。

**図** 11-3　**デファードライト処理の並列** WRITE **機能の概要**




## (1) デファードライト処理の並列 WRITE 機能の効果

書き込み処理を複数の並列処理用デファード WRITE プロセスで実行するため，ディスクへの書き込み時間が短縮されます。

## (2) 指定方法

pd_dfw_awt_process オペランドに書き込み処理をする並列処理用デファード WRITE プロセス数を指定します。また，デファードライトトリガの要求比率を pd_dbbuff_rate_updpage オペランドに指定します。なお，pd_dfw_awt_process オペランドの記述がない場合，デファードライト処理の並列 WRITE 機能は無効になります。

## (3) 指定上の考慮点

デファードライト処理の並列 WRITE 機能を指定すると，並列処理用デファード WRITE プロセス数が増加するため，CPU 利用率が上がります。

# 11.7 コミット時反映処理の設定

**コミット時反映処理**とは，グローバルバッファ上で更新されたページを COMMIT 文発行時にディスクに書き込む処理のことです。

## (1) コミット時反映処理の効果

COMMIT 文の発行時にデータベースの更新内容をディスクに書き込むため，トランザクションの完了時点でデータベースの内容が保証されます。そのため，全面回復処理時に，シンクポイント時点からデータベースを回復する必要がなく，全面回復処理の時間が短縮できます。

## (2) 指定方法

pd_dbsync_point オペランドに commit を指定します。

ただし，LOB 用 RD エリアはこのオペランドの影響を受けません。ディレクトリ部は COMMIT 文発行時点で反映されます。データ部は LOB 用グローバルバッファを割り当てているかどうかによって処理が異なります。LOB 用グローバルバッファを割り当てていない場合は更新要求時にすぐに反映されます。
LOB 用グローバルバッファを割り当てている場合は COMMIT 文発行時点で反映されます。ただし，グローバルバッファが満杯になったときはその時点で反映されます。

## (3) 指定上の考慮点

pdbufls コマンドで取得した情報を参照して，ディスクへの出力回数が多くて更新要求ヒット率が低い場合には，グローバルバッファのバッファ面数を大きく設定します。

# 11.8 グローバルバッファの LRU 管理方式

HiRDB では，業務の種類（オンライン業務又はバッチ業務）によって，**グローバルバッファの** LRU **管理方式**を選択できます。

## 11.8.1 LRU の管理方式

LRU の管理方式には，次に示す 2 種類があります。

- 参照バッファ及び更新バッファの独立した LRU での管理
- グローバルバッファの一括した LRU での管理

### (1) 参照バッファ及び更新バッファの独立した LRU での管理

参照バッファ及び更新バッファをそれぞれ独立した LRU で管理します。

グローバルバッファの不足時には，グローバルバッファ内のアクセスした参照バッファの中で，最も古いバッファがメモリから追い出されます。

#### (a) 適用基準

参照バッファ及び更新バッファをそれぞれ独立した LRU で管理した方がよい場合を次に示します。

- 検索処理に比べて，更新処理の割合が比較的少ない場合で更新バッファヒット率が高い場合（オンライン業務のように 1 トランザクション当たりの参照，更新件数が比較的少ない場合）

#### (b) 指定方法

システム共通定義の pd_dbbuff_lru_option オペランドに SEPARATE を指定します。

#### (c) 注意事項

- 大量の更新処理が発生した場合，参照バッファヒット率が低下し，検索処理が遅くなります。
- 次のどちらかに該当する場合は，無条件に pd_dbbuff_lru_option オペランドに MIX を仮定します。そのため，参照バッファ及び更新バッファの独立した LRU での管理はできません。
  - pd_dbsync_point オペランドに commit を指定している場合
  - pd_dbbuff_binary_data_lru オペランドに N を指定している場合

### (2) グローバルバッファの一括した LRU での管理

グローバルバッファを一括した LRU で管理します。グローバルバッファの不足時には，グローバルバッファ内のアクセスしたバッファで，最も古いバッファがメモリから追い出されます。

#### (a) 適用基準

グローバルバッファを一括した LRU で管理した方がよい場合を次に示します。

- 検索処理に比べて，更新処理の割合が多い場合，突発的な大量検索又は大量更新が発生する場合（オンライン業務とバッチ業務など，大量検索，大量更新が共存する場合）

#### (b) 指定方法

システム共通定義の pd_dbbuff_lru_option オペランドに MIX を指定します。

システム共通定義の pdbuffer オペランドの -w オプションに，デファードライトトリガでの更新ページの出力比率を指定します。

**(c) 注意事項**

- 更新バッファヒット率が高い場合には，大量検索によって更新バッファが一時的にメモリから追い出されます。このような場合には，更新処理の延長でファイルの読み込みが発生し，処理が遅くなる場合があります。
- pd_dbsync_point=sync の指定又は省略時には，検索処理の延長でファイルへの書き込みが発生し，検索処理が遅くなる場合があります。

## 11.8.2 UAP ごとの LRU 管理抑止設定

OLTP 環境で，UAP の大量検索や大量更新によってグローバルバッファにキャッシュされた直近の内容がメモリから追い出され，OLTP 性能を一時的に低下させてしまうことがあります。このとき，大量検索や大量更新をする UAP を特定できるのであれば，UAP ごとに LRU 管理を抑止する設定をすることで，OLTP 性能の低下を回避できます。

**参考**

LRU 管理を抑止できるのは UAP からのアクセスだけです。コマンドやユティリティからのアクセスの場合は，LRU 管理が行われます。ただし，次のコマンドは LRU 管理が抑止されます。

- pddbst
  -b オプションを指定して LRU 管理を抑止できます。

### (1) 適用基準

OLTP 環境で，グローバルバッファを使用して大量検索や大量更新をする UAP を実行する場合に適用することをお勧めします。

### (2) 適用の効果

LRU 管理を抑止した UAP がアクセスしたページはアクセス頻度に関係なく，最も古い時にアクセスしたページとしてグローバルバッファ上にキャッシュされます。そのため，LRU 管理を適用した UAP がアクセスしたページより先にメモリから追い出されるようになり，LRU 管理を適用した UAP がアクセスしたページはメモリから追い出されなくなります。ただし，LRU 管理を抑止する UAP と LRU 管理を抑止しない UAP が同じページにアクセスする場合，そのページは LRU 管理されます。

### (3) 指定方法

クライアント環境定義の PDDBBUFLRU オペランドに NO を指定します。

### (4) 注意事項

1. LRU 管理を抑止する UAP がアクセスしたページは，アクセス頻度に関係なく，バッファ不足が発生すると，メモリからの追い出し対象となります。そのため，LRU 管理を抑止する UAP はバッファヒット率の低下に伴う入出力回数の増加によって，レスポンス性能が低下することがあります。
2. UAP は 1 ～ 4 面のバッファを同時に確保します。そのため，LRU 管理を抑止する設定をした場合でも，UAP ごとにグローバルバッファにキャッシュされているページの中で 1 ～ 4 ページはキャッシュから追い出されるおそれがあります。
3. 更新を行う UAP に対して LRU 管理を抑止した場合，DB への書き込み回数が多く，ログ出力契機が頻繁に発生するため，LRU 管理を抑止しない場合に比べて，出力されるログ量が多くなります。容量が不足しないように次のようにしてください。
   - システムログファイルの容量を再度見積もりする

- ログレスモードで実行できる場合，クライアント環境定義の PDDBLOG に NO を指定する

LRU 管理を抑止する場合のログ量は，次に示す計算式で求めます。なお，pd_log_rec_leng オペランドの指定値を 1024 にすると，LRU 管理を抑止する場合に出力されるログ量を最小に抑えられます。

| |
|---|
| 更新 GET 数※× pd_log_rec_leng オペランドの値 |

注※

更新 GET 数は，UAP 統計レポートの DIDUC の値，又は UAP に関する統計情報の DIDUC の値で確認できます。

4. LRU 管理を抑止する UAP でも，ロールバックが発生した場合は LRU 管理されることがあります。LRU 管理されるかどうかはロールバックのタイミングによって次のように異なります。

| ロールバックのタイミング | | LRU 管理されるかどうか |
|---|---|---|
| SQL でタイミングを定義する場合 | 制御系 SQL の ROLLBACK 文を指定して，ROLLBACK 文が実行される直前のコミット時点までロールバックされるとき | LRU 管理は抑止されます。 |
| HiRDB によって自動的にロールバックされる場合 | SQL 実行時に処理が続行できなくなり，HiRDB によって直前のコミット時点まで暗黙的にロールバックされるとき | LRU 管理は抑止されます。 |
| | UAP の異常終了時に HiRDB によって直前のコミット時点までロールバックされるとき | LRU 管理が行われます。 |
| | ディレードリランによってロールバックされるとき | LRU 管理が行われます。 |

## 11.8.3 UAP がアクセスするバイナリデータの LRU 管理抑止設定

サイズの大きなバイナリデータを大量にアクセスする UAP を実行する場合，バイナリデータがグローバルバッファにキャッシュされると，グローバルバッファにキャッシュされた直近の内容がメモリから追い出され，性能が一時的に低下することがあります。このとき，バイナリデータのアクセス頻度が低い場合は，バイナリデータの分岐行ページの LRU 管理だけを抑止することで，性能の低下を回避できます。

**参考**

LRU 管理を抑止できるのは UAP からのアクセスだけです。コマンドやユティリティからのアクセスの場合は，LRU 管理が行われます。ただし，次のコマンドは LRU 管理が抑止されます。

- プラグインが提供するコマンド
- pddbst
  -b オプションを指定して LRU 管理を抑止できます。

また，次のコマンドの場合，LRU 管理は抑止できませんが，グローバルバッファにキャッシュされた直近の内容がメモリから追い出されることを回避できます。

- pdload，pdrorg，pdrbal
  -n オプションを指定してローカルバッファを使用すると，グローバルバッファから基本行内のデータが追い出されるのを回避できます。
- pdpgbfon
  -b オプションを省略すると，バイナリデータの分岐行ページをアクセスしません。

### (1) 適用基準

次のすべての条件を満たす場合に適用することをお勧めします。

- BINARY 型や，BINARY 型の属性を含む抽象データ型，XML 型などのサイズの大きなバイナリデータを含む表がある場合
- バイナリデータへのアクセスはまれである場合

**注意事項**

頻繁にバイナリデータをアクセスする場合には，適用しないでください。

## (2) 適用の効果

バイナリデータが格納された分岐行はアクセス頻度に関係なく，最も古い時にアクセスしたページとしてグローバルバッファ上にキャッシュされます。そのため，分岐行ページが基本行ページより先にメモリから追い出されるようになり，基本行内のデータがメモリから追い出されなくなります。

## (3) 指定方法

システム共通定義の pd_dbbuff_binary_data_lru オペランドに N を指定します。

## (4) 注意事項

1. LRU 管理を抑止した場合，UAP がアクセスするバイナリデータの分岐行ページは，アクセス頻度に関係なく，バッファ不足が発生するとメモリからの追い出し対象となります。そのため，バイナリデータの分岐行をアクセスする UAP はバッファヒット率の低下に伴う入出力回数の増加によって，レスポンス性能が低下することがあります。
2. UAP は 1 ～ 4 面のバッファを同時に確保します。そのため，LRU 管理を抑止する設定をした場合でも，UAP ごとにグローバルバッファにキャッシュされているページの中で 1 ～ 4 ページはキャッシュから追い出されるおそれがあります。
3. LRU 管理を抑止した場合，バイナリデータの分岐行ページの更新を行う UAP を実行するときに DB への書き込み回数が多く，ログ出力契機が頻繁に発生します。そのため，LRU 管理を抑止しない場合に比べて，出力されるログ量が多くなります。容量が不足しないように次のようにしてください。
   - システムログファイルの容量を再度見積もる
   - ログレスモードで実行できる場合，クライアント環境定義の PDDBLOG に NO を指定する

   LRU 管理を抑止する場合のログ量は，次に示す計算式で求めます。なお，pd_log_rec_leng オペランドの指定値を 1024 にすると，LRU 管理を抑止する場合に出力されるログ量を最小に抑えられます。

バイナリデータをアクセスする UAP の更新 GET 数※× pd_log_rec_leng オペランドの値

   注※

   更新 GET 数は，UAP 統計レポートの DIDUC の値，又は UAP に関する統計情報の DIDUC の値で確認できます。
4. LRU 管理を抑止する UAP でも，ロールバックが発生した場合は LRU 管理されることがあります。LRU 管理されるかどうかはロールバックのタイミングによって次のように異なります。

| ロールバックのタイミング | | LRU 管理されるかどうか |
|---|---|---|
| SQL でタイミングを定義する場合 | 制御系 SQL の ROLLBACK 文を指定して，ROLLBACK 文が実行される直前のコミット時点までロールバックされるとき | LRU 管理は抑止されます。 |

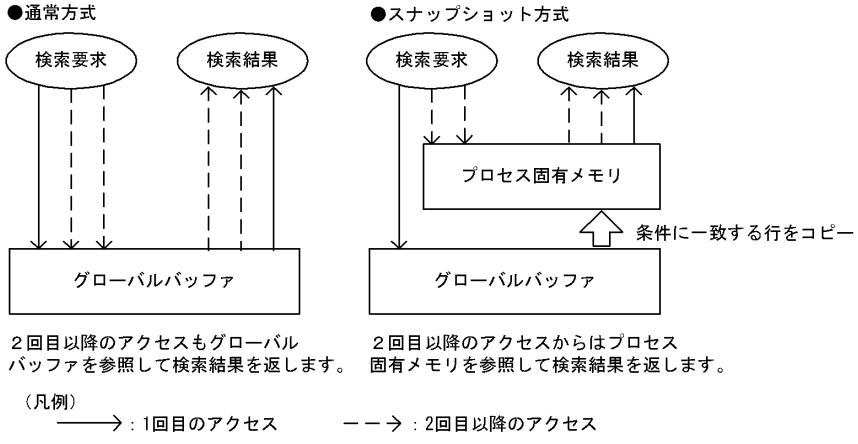
| ロールバックのタイミング | | LRU 管理されるかどうか |
|---|---|---|
| HiRDB によって自動的にロールバックされる場合 | SQL 実行時に処理が続行できなくなり，HiRDB によって直前のコミット時点まで暗黙的にロールバックされるとき | LRU 管理は抑止されます。 |
| | UAP の異常終了時に HiRDB によって直前のコミット時点までロールバックされるとき | LRU 管理が行われます。 |
| | ディレードリランによってロールバックされるとき | LRU 管理が行われます。 |

5. この機能を適用した場合，無条件に pd_dbbuff_lru_option オペランドに MIX を仮定します。そのため，参照バッファ及び更新バッファの独立した LRU での管理はできません。

# 11.9 スナップショット方式によるページアクセス

性能向上を目的とした機能（グループ分け高速化機能など）を適用できない検索をするとき，条件に合致する行数とほぼ同数回グローバルバッファにアクセスしています。**スナップショット方式**では，最初のアクセス時にバッファ内の探索条件に一致するすべての行をプロセス固有メモリ上にコピーし，2 回目以降の同一ページのアクセスはプロセス固有メモリ上を参照して検索結果を返します。スナップショット方式の概要を次の図に示します。

**図 11-4 スナップショット方式の概要**




## (1) スナップショット方式によるアクセスの効果

最初に探索条件に一致する行をプロセス固有メモリにコピーするため，2 回目以降のアクセス時に掛かる検索時間を短縮できます。また，グローバルバッファへのアクセス回数を削減し，同一グローバルバッファへのアクセスの集中を防ぎます。

## (2) 指定方法

pd_pageaccess_mode オペランドに SNAPSHOT（省略値）を指定します。

## (3) 指定上の考慮点

スナップショット方式を指定すると，表又はインデクスの格納 RD エリアのページサイズに基づいて，動的にプロセス固有メモリが確保されます。確保されるプロセス固有メモリの計算式については，HiRDB/シングルサーバの場合「15.1.6(4) スナップショット方式指定時に必要なメモリ所要量の求め方」を参照してください。HiRDB/ パラレルサーバの場合「15.2.6(5) スナップショット方式指定時に必要なメモリ所要量の求め方」を参照してください。

## (4) スナップショット方式の適用可否

検索時にスナップショット方式を適用するかどうかを次の表に示します。

**表 11-1 検索時のスナップショット方式の適用可否**

| 条件 | | アクセス対象別の適用可否 | |
|---|---|---|---|
| | | 表 | インデクス |
| システム共通定義に pd_indexlock_mode=KEY を指定した場合に次の条件を満たさない検索<br>• WITHOUT LOCK NOWAIT 指定の検索<br>• LOCK TABLE 前提の検索 | | － | × |
| ホールダブルカーソルを使った検索 | | × | × |
| 検索方式がインデクススキャン（INDEX SCAN，MULTI COLUMNS INDEX SCAN）の検索 | WITHOUT LOCK WAIT 指定の場合 | × | × |
| | 上記以外 | × | ○ |
| 検索方式が ROWID FETCH の検索 | | × | － |
| 次の列を指定している検索<br>• 定義長 256 バイト以上の VARCHAR，MVARCHAR，NVARCHAR 型の列<br>• 繰返し列<br>• 抽象データ型の列<br>• LOB 列<br>• 定義長 256 バイト以上の BINARY 型の列 | WITHOUT LOCK WAIT 指定の場合 | × | △ |
| | 上記以外 | × | ○ |
| ページ内のヒット件数が 1 件になるような探索条件を指定した検索 | | × | × |
| プラグインインデクスを使った検索 | | × | × |
| ディクショナリ表に対する検索 | | × | × |

（凡例）

○：適用します。

×：適用しません。

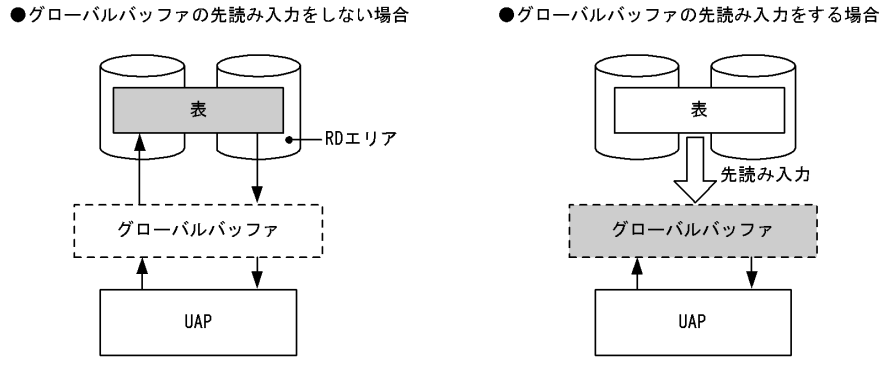
△：検索方式がキースキャン（KEY SCAN，MULTI COLUMNS KEY SCAN）の場合に適用します。

－：条件に依存しません。又は該当しません。

# 11.10 グローバルバッファの先読み入力

**グローバルバッファの先読み入力**とは，指定した表やインデクスのデータをあらかじめグローバルバッファに読み込みしておく機能です。概要を次の図に示します。

**図 11-5 グローバルバッファの先読み入力の概要**




〔説明〕

- グローバルバッファの先読み入力をしない場合
  HiRDB 開始直後に UAP が表にアクセスする時，グローバルバッファにはデータがないため，表からデータを読み込みます（物理的な入出力が発生します）。以降，この表のデータにアクセスする時は，グローバルバッファに読み込まれているページについては表からの読み込みは発生しません。ただし，ほかのページのデータにアクセスする時は，読み込み処理が発生します。
- グローバルバッファの先読み入力をする場合
  あらかじめ表のデータをグローバルバッファに読み込んでいるため，表からデータを読み込まないで表にアクセスできます（物理的な入出力は発生しません）。以降，この表にアクセスする時，表からの読み込みはありません。

## (1) グローバルバッファの先読み入力の効果

指定した表やインデクスのデータを先読み入力しておくため，バッファヒット率が高くなります。HiRDB 開始直後，オンライン業務開始前などに，入出力が多いと思われる表やインデクスを先読みしておくことで，高いバッファヒット率が期待できます。

## (2) 実行方法

先読み入力する表やインデクスを指定し，グローバルバッファ常駐化ユティリティ（pdpgbfon）を実行します。

## (3) 使用上の考慮点

- グローバルバッファの面数は，先読み入力する表やインデクスが格納されているページ数より多く必要です。
- グローバルバッファの面数が十分でない場合，LRU 管理方式によってグローバルバッファから古いページ情報が追い出されます（システム定義の pd_dbbuff_lru_option オペランドの値に従って，アクセスしたグローバルバッファ中の最も古いページが追い出されます）。そのため，グローバルバッファの面数が十分でない場合，pdpgbfon を実行しても意味がありません。
- グローバルバッファ常駐化ユティリティ（pdpgbfon）で先読みする場合，格納ページ順の読み込みとなるため，プリフェッチ機能が有効となります。グローバルバッファを定義する場合，プリフェッチ数を

指定することで実行時間の短縮が図れます。

# 11.11 ローカルバッファ

**ローカルバッファ**とは，ディスク上の RD エリアに格納されているデータを入出力するためのバッファのことで，プロセス固有メモリ上に確保されます。ローカルバッファには次に示す種類があります。

- **インデクス用ローカルバッファ**
  インデクスデータの入出力に使用されるローカルバッファです。インデクス用ローカルバッファはインデクス単位に割り当てます。
- **データ用ローカルバッファ**
  データの入出力に使用されるローカルバッファです。データ用ローカルバッファは RD エリア単位に割り当てます。

ローカルバッファは，UAP ごとに UAP **環境定義**で定義できます。UAP に専用のローカルバッファを割り当てることで，他 UAP によるグローバルバッファの占有やバッファの排他処理による待ち状態を避けられます。UAP 環境定義の詳細は，マニュアル「HiRDB Version 8 システム定義」を参照してください。

次に示す条件をすべて満たす場合にローカルバッファを定義します。

- 大量のデータを検索又は更新する
- アクセス対象の RD エリアがほかの UAP からアクセスされない

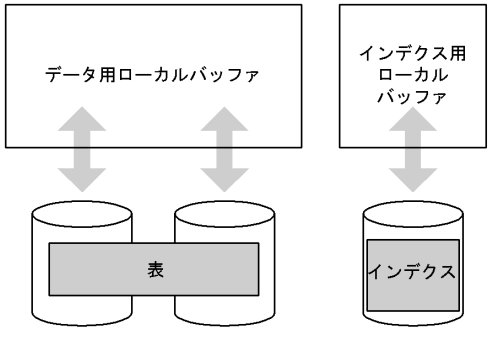
なお，HiRDB に常時接続する UAP はシステムへの影響（メモリの圧迫，サーバプロセスの占有など）が大きいため，ローカルバッファを定義しないでください。

## 11.11.1 インデクス用ローカルバッファの割り当て

データ用ローカルバッファとインデクス用ローカルバッファを分けて定義すると，データの検索とインデクスの検索を同時に実行しても互いが独立して動作します。そのため，大量データの全件検索時などでもインデクスの入出力回数を削減でき，処理時間を短縮できます。

表データとインデクスデータそれぞれにローカルバッファを割り当てた場合のインデクス用ローカルバッファの概要を次の図に示します。

**図 11-6 インデクス用ローカルバッファの概要**




## 11.11.2 データ用ローカルバッファの割り当て

### (1) 異なるページ長の RD エリアが複数ある場合

異なるページ長の RD エリアが複数ある場合は，同じか又は近いページ長の RD エリアをまとめて一つの

ローカルバッファに割り当てます。これによって，メモリの使用効率が良くなります。

異なるページ長の RD エリアをまとめて一つのローカルバッファに割り当てると，ページ長が最も大きい RD エリアに合わせてローカルバッファが確保されます。このため，ページ長が小さい RD エリアに対してデータページの入出力をした場合，1 面のローカルバッファに使用されない領域ができることになり，メモリの使用効率が下がります。

### （2）UAP からのアクセス方法が異なる RD エリアが複数ある場合

同じページ長でも，UAP からのアクセス方法が異なる RD エリアが複数ある場合，例えば，用途が異なる RD エリア，順次処理が多くて更新が少ない RD エリア，追加や更新が多い RD エリアなどがある場合は，それぞれ異なるローカルバッファを割り当てるようにします。

## 11.11.3 ローカルバッファの割り当て方法

インデクス用ローカルバッファを割り当てる場合，pdlbuffer オペランドの -i オプションでインデクス用のローカルバッファを割り当てるインデクスの名称（認可識別子．インデクス識別子）を指定します。

データ用ローカルバッファを割り当てる場合，pdlbuffer オペランドの -r オプションでデータ用のローカルバッファを割り当てる RD エリアの名称を指定します。

ローカルバッファの定義例を次に示します。

```
pdlbuffer -a localbuf01 -r RDAREA01,RDAREA02 -n 1000       1
pdlbuffer -a localbuf02 -i USER01.INDX01 -n 1000           2
```

〔説明〕

1. RD エリア（RDAREA01，RDAREA02）にデータ用ローカルバッファを割り当てます。
2. インデクス（USER01.INDX01）にインデクス用ローカルバッファを割り当てます。

## 11.11.4 ローカルバッファ使用時の注意

ローカルバッファの使用時にサーバプロセスが異常終了すると，アボートコード Phb3008 を出力して HiRDB（HiRDB/ パラレルサーバの場合はユニット）が異常終了することがあります。サーバプロセスの異常終了時に更新ページがあると，トランザクション回復プロセスで回復処理ができないことがあります。その場合，HiRDB の再開始時に回復処理を行います。ローカルバッファを使用している場合に障害が発生したときの HiRDB の処理と対処方法については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

# 12 表の設計

この章では，表を設計する上で検討する項目について説明します。


12.1 **表を設計するときの検討項目**

12.2 **表の正規化**

12.3 **表の横分割**

12.4 **表のマトリクス分割**

12.5 **トリガの定義**

12.6 **ビュー表の作成**

12.7 FIX **属性の指定**

12.8 **主キー（プライマリキー）の指定**

12.9 **クラスタキーの指定**

12.10 **サプレスオプションの指定**

12.11 **ノースプリットオプションの指定**

12.12 **バイナリデータ列の指定**

12.13 **文字集合の指定**

12.14 WITHOUT ROLLBACK **オプションの指定**

12.15 **改竄防止機能の指定**

12.16 **繰返し列を含む表**

12.17 **抽象データ型を含む表**

12.18 **共用表**

12.19 **参照制約**

12.20 **検査制約**

# 12.1 表を設計するときの検討項目

HiRDB のデータベースはリレーショナルデータベースです。その論理構造である表をどのように設計するかを検討します。

まず，表を**正規化**しておくことが必要です。ただし，同じように正規化した表であっても，ユーザ用 RD エリアへの格納の仕方などによって，表に対する処理性能が異なります。また，処理性能よりも操作性を重視する場合もあるため，期待する効果を考慮した表の設計が必要になります。表を設計するときの検討項目を次の表に示します。

**表 12-1 データベースの表を設計するときの検討項目**

| 設計作業ごとの検討項目 | | 長所 | 短所 | 記載箇所 |
|---|---|---|---|---|
| 表の正規化 | | 表の格納効率や処理効率が向上します。 | 表の検索時に，正規化後の表同士の結合検索が必要になる場合は，処理性能が落ちることがあります。 | 12.2 |
| 表の横分割 | 表の横分割の指定 | • RD エリアごとの運用ができます。<br>• HiRDB/ パラレルサーバの場合は，表にアクセスする処理を複数の RD エリアにわたって並列化できるため，表に対するアクセスの高速化及び負荷の分散が図れます。 | • RD エリアの数が増えます。<br>• この表に作成するインデクスを横分割しないと，インデクスでの排他制御によって，同時実行性が低下することがあります。 | 12.3 |
| | キーレンジ分割 | • 表のデータをどの RD エリアに格納したかが分かります。<br>• 各業務に関連する部分のデータを別々の RD エリアにまとめて格納し，RD エリアごとの運用ができます。 | キーレンジを意識しないと，RD エリアに均等に格納できません。 | |
| | フレキシブルハッシュ分割※ | • キーレンジを意識しなくても，データを均等に RD エリアに格納できます。<br>• RD エリア，ハッシュ関数を変更しやすくなります。<br>• CPU，ディスクの追加に対応できます。 | • 表のデータをどの RD エリアに格納したかが分かりません。<br>• 特定のキーに偏りや重複があるとデータが均等に格納できません。<br>• キーの一意性のチェックができません。 | |
| | FIX ハッシュ分割※ | • キー値によって , データを格納する RD エリアが一意に決定されます。<br>• キーレンジを意識しなくても，データを均等に RD エリアに格納できます。<br>• CPU，ディスクを追加しやすくなります。<br>• 表分割ハッシュ関数を使用した UAP を作成し，入力データを RD エリアごとに格納できます。 | 表にデータが格納されているとユーザ用 RD エリアの追加及びハッシュ関数の変更ができません。 | |
| 表のマトリクス分割 | | • キーレンジ分割したデータを，さらに別の列値で分割できるため，通常のキーレンジ分割よりも，SQL の高速処理，運用時間の短縮が期待できます。<br>• キーレンジ分割とハッシュ分割の組み合わせで分割できるため，適用業務の幅が広がります。 | 通常の横分割に比べ，RD エリアを更に細分化できるため，運用や管理が複雑になります。 | 12.4 |
| トリガの定義 | | ある表への操作を契機として，自動的に SQL 文を実行するようにできます。 | 特にありません。 | 12.5 |

| 設計作業ごとの検討項目 | 長所 | 短所 | 記載箇所 |
|---|---|---|---|
| ビュー表の作成 | • ほかのユーザに実表のアクセス権限を与えないで，ビュー表だけのアクセス権限を与えると，アクセス可能な実表の範囲を行又は列単位で制限できます。<br>• 複雑な問い合わせ指定をした場合に検索できるデータであらかじめビュー表を作成すると，表を参照する操作が手軽になります。<br>• ビュー表を通して実表を参照又は更新できます。<br>• ビュー表を通して外部表を参照，更新できます。 | 特にありません。 | 12.6 |
| FIX 属性の指定 | • 行単位インタフェースを使用する場合は列数が多くてもアクセス性能を向上できます。<br>• FIX 属性の表に対する入力データにナル値を許さないように制約できます。<br>• 列数の多い表の場合はディスク所要量を削減できます。 | 特にありません。 | 12.7 |
| 主キー（プライマリキー）の指定 | 主キーを定義した列には，一意性制約と非ナル値制約が適用されます。 | 特にありません。 | 12.8 |
| クラスタキーの指定 | • 範囲を指定した行の検索，更新，削除などをするときやクラスタキー順の検索，更新などをするときに入出力時間を削減できます。<br>• クラスタキーに UNIQUE を指定すると，クラスタキーの構成列の値がすべての行で重複しないように制約できます。<br>• 表を作成するときの入力データがクラスタキーの昇順又は降順に並んでいるかどうかをデータベース作成ユティリティ（pdload）で確認できます。<br>• 表を再編成するときに，アンロードした行のクラスタキーと，リロードするクラスタキーが一致しているかどうかをデータベース再編成ユティリティ（pdrorg）で確認できます。 | • クラスタキーを構成する列の値を更新できません。<br>• クラスタキーを構成する列の値にナル値を挿入できません。<br>• クラスタキーを指定した表にデータを追加するとき，追加しようとするキー値に近接するキー値を持ったページを探すためのオーバヘッドが発生します。 | 12.9 |
| サプレスオプションの指定 | • ディスク所要量を削減できます。<br>• 全件検索などの検索処理での入出力時間を削減できます。 | 特にありません。 | 12.10 |
| ノースプリットオプションの指定 | データの格納効率を向上できるため，ディスク所要量を削減できます。 | 特にありません。 | 12.11 |
| バイナリデータ列の指定 | 文書，画像，音声などの可変長データを指定できます。 | 特にありません。 | 12.12 |
| 文字集合の指定 | 表の列ごとに異なる文字集合の文字列データを格納できます。これによって，次のことができます。<br>• VOS3 システムから HiRDB に移行した場合に，データベースに格納していた文字データを VOS3 システムの文字列データの照合順で検索，代入，比較ができます。<br>• UTF-16 で文字データの検索，代入，比較ができます。 | 特にありません。 | 12.13 |
| WITHOUT ROLLBACK オプションの指定 | 採番業務で使用する表を定義する場合，表に対する更新処理完了を契機に更新行に対する排他が解除されるため，排他待ちの発生を削減できます。 | 特にありません。 | 12.14 |

| 設計作業ごとの検討項目 | 長所 | 短所 | 記載箇所 |
|---|---|---|---|
| 改竄防止機能の指定 | 表データを誤って，又は不当に更新されることを防止できます。 | 改竄防止表がある RD エリアに対して使用できない機能や，SQL，ユティリティ，コマンドの実行に制限があります。 | 12.15 |
| 繰返し列を含む表 | • 複数の表の結合が不要になります。<br>• 重複する情報がなくなるため，ディスク容量を削減できます。<br>• 関連データ（繰返しデータ）が近くに格納されるため，別の表にするよりもアクセス性能が優れています。 | 特にありません。 | 12.16 |
| 抽象データ型を含む表 | 複雑な構造のデータを表に格納して，通常の表データと同様に操作できます。 | 特にありません。 | 12.17 |
| 共用表 | • 複数バックエンドサーバ間の接続やデータ転送によるオーバヘッドが削減できます。<br>• トランザクションの多重実行時など，並列処理の効率が上がります。 | 共用表を更新する場合，全バックエンドサーバで，更新する共用表がある RD エリアに排他を掛けるので，共用 RD エリアのほかの表にアクセスする業務があると，デッドロックが発生することがあります。 | 12.18 |
| 参照制約 | 複数の表間のデータの整合性チェック，及びデータ操作を自動化できます。 | 被参照表や参照表を更新する場合，データの整合性をチェックするため，チェックに掛かる処理時間が増加します。 | 12.19 |
| 検査制約 | データの追加又は更新時のチェックを自動化できます。 | 検査制約が定義された表を更新する場合，データの整合性をチェックするため，チェックに掛かる処理時間が増加します。 | 12.20 |

注※

次に示す場合は，**ハッシュ分割表のリバランス機能**を使用することをお勧めします。

- 表をハッシュ分割する場合
- データ量の増加が見込まれる場合

ハッシュ分割表のデータ量が増加したため RD エリアを追加すると（表の横分割数を増やすと），既存の RD エリアと新規追加した RD エリアとの間でデータ量の偏りが生じます。ハッシュ分割表のリバランス機能を使用すると，表の横分割数を増やすときにデータ量の偏りを修正できます。ハッシュ分割表のリバランス機能については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

# 12.2 表の正規化

表を正規化することは，表の格納効率や処理効率の向上を図る上で重要です。表を正規化する際は，表の構成列を検討します。ここでは，次に示す正規化について説明します。

- 表の格納効率を向上させる正規化
- 表の処理効率を向上させる正規化

## (1) 表の格納効率を向上させる正規化

一つの表に同じような情報を持つ列が複数ある場合は，この表を複数の表に分けて，それぞれの表に同じような情報を持つ列がなくなるように正規化します。これによって，表に対するデータの格納効率が良くなります。これを次の図の例で説明します。

**図 12-1　表に同じような情報を持つ列が複数ある場合**

● 正規化前

ZAIKO

| 商品番号 | 商品コード | 商品名 | 単価 | 在庫量 |
|---|---|---|---|---|
| SNO | SCODE | SNAME | TANKA | SURYO |
| 01010 | 101 | ブラウス | 3500 | 62 |
| 01011 | 101 | ブラウス | 3500 | 85 |
| 02021 | 202 | ポロシャツ | 3640 | 67 |
| 03530 | 353 | スカート | 4760 | 18 |
| 03531 | 353 | スカート | 4760 | 56 |
| 04121 | 412 | セーター | 8400 | 22 |
| 05910 | 591 | ソックス | 250 | 300 |
| 05911 | 591 | ソックス | 250 | 90 |
| 05912 | 591 | ソックス | 250 | 280 |
| 06710 | 671 | トレーナー | 4500 | 45 |
| 06711 | 671 | トレーナー | 4500 | 76 |

列が１対１に対応しています。
列の情報が冗長になっています。

● 正規化後

ZAIKO

| SNO | SNAME | TANKA | SURYO |
|---|---|---|---|
| 01010 | ブラウス | 3500 | 62 |
| 01011 | ブラウス | 3500 | 85 |
| 02021 | ポロシャツ | 3640 | 67 |
| 03530 | スカート | 4760 | 18 |
| 03531 | スカート | 4760 | 56 |
| 04121 | セーター | 8400 | 22 |
| 05910 | ソックス | 250 | 300 |
| 05911 | ソックス | 250 | 90 |
| 05912 | ソックス | 250 | 280 |
| 06710 | トレーナー | 4500 | 45 |
| 06711 | トレーナー | 4500 | 76 |

SHOHIN

| SCODE | SNAME |
|---|---|
| 101 | ブラウス |
| 202 | ポロシャツ |
| 353 | スカート |
| 412 | セーター |
| 591 | ソックス |
| 671 | トレーナー |

〔説明〕

正規化前の ZAIKO 表の SCODE の列と SNAME の列は 1 対 1 で対応し，それぞれの列の情報は冗長になっています。このような場合は，ZAIKO 表から，SCODE の列と SNAME の列で構成される SHOHIN 表をほかに作成します。このとき，SHOHIN 表では，SCODE の列と SNAME の列に重複した情報を持たないようにします。

## (2) 表の処理効率を向上させる正規化

### (a) 複数の業務で同一の表を使用する場合

複数の業務で同一の表を使用する場合は，それぞれの業務で使用する列によって，この表を業務ごとの表

に正規化します。これによって，それぞれの表に対する同時実行性が向上します。これを次の図の例で説明します。

**図 12-2　複数の業務で同一の表を使用する場合**

● 正規化前

SKANRI（商品管理表）

| 商品番号 | 商品名 | 標準単価 | 在庫量 | 受注量 | 受注単価 |
|---|---|---|---|---|---|
| SNO | SNAME | HTANKA | ZSURYO | JSURYO | JTANKA |
| 01010 | ブラウス | 3500 | 62 | 10 | 3150 |
| 01011 | ブラウス | 3500 | 85 | 8 | 3150 |
| 02021 | ポロシャツ | 3640 | 67 | 13 | 3280 |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |

更新 在庫管理　　更新 受注管理

● 正規化後

ZAIKO

| SNO | SNAME | HTANKA | ZSURYO |
|---|---|---|---|
| 01010 | ブラウス | 3500 | 62 |
| 01011 | ブラウス | 3500 | 85 |
| 02021 | ポロシャツ | 3640 | 67 |
| ⋮ | ⋮ | ⋮ | ⋮ |

更新 在庫管理

JUTYU

| SNO | SNAME | JSURYO | JTANKA |
|---|---|---|---|
| 01010 | ブラウス | 10 | 3150 |
| 01011 | ブラウス | 8 | 3150 |
| 02021 | ポロシャツ | 13 | 3280 |
| ⋮ | ⋮ | ⋮ | ⋮ |

更新 受注管理

〔説明〕

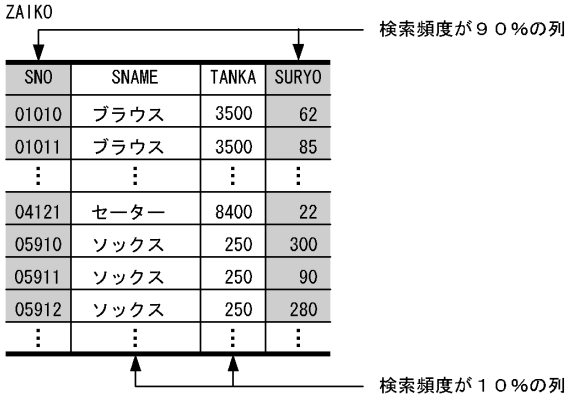
正規化前の SKANRI 表（商品管理表）を在庫管理業務と受注管理業務で使用しています。このような場合は，SKANRI 表を在庫管理業務だけで使用する ZAIKO 表と，受注管理業務だけで使用する JUTYU 表に正規化します。

### (b) アクセス頻度の高い列と低い列がある場合

一つの表に，アクセス頻度が高いと考えられる列と低いと考えられる列がある場合は，この表をアクセス頻度が高い列で構成した表と，アクセス頻度が低い列で構成した表に正規化します。これを次の図の例で説明します。

**図 12-3　アクセス頻度の高い列と低い列がある場合**

● 正規化前




● 正規化後

ZAIKO2

| SNO | SURYO |
|---|---|
| 01010 | 62 |
| 01011 | 85 |
| ⋮ | ⋮ |
| 04121 | 22 |
| 05910 | 300 |
| 05911 | 90 |
| 05912 | 280 |
| ⋮ | ⋮ |

SHOHIN

| SNO | SNAME | TANKA |
|---|---|---|
| 01010 | ブラウス | 3500 |
| 01011 | ブラウス | 3500 |
| ⋮ | ⋮ | ⋮ |
| 04121 | セーター | 8400 |
| 05910 | ソックス | 250 |
| 05911 | ソックス | 250 |
| 05912 | ソックス | 250 |
| ⋮ | ⋮ | ⋮ |

〔説明〕

正規化前の ZAIKO 表を検索する場合，SNO 及び SURYO の列と，SNAME 及び TANKA の列の間で，検索頻度の比率が 9：1 であるとします。このような場合は，ZAIKO 表を検索頻度の高い列の集まり（ZAIKO2 表）と検索頻度の低い列の集まり（SHOHIN 表）の二つの表に正規化します。

例えば，ZAIKO 表を全件検索するのに，10000 回の物理的な入出力が必要であるとします。ZAIKO 表を ZAIKO2 表と SHOHIN 表に分割したことで，ZAIKO2 表の検索が 4500 回（=5000 × 0.9），SHOHIN 表の検索が 500 回（=5000 × 0.1），合計 5000 回の物理的な入出力で済むことになり，全体としての表の処理効率が向上します。

# 12.3　表の横分割

ここでは，表の横分割の設計方法について説明します。

## 12.3.1　表の横分割の概要

一つの表を複数のユーザ用 RD エリアに分割して格納することを**表の横分割**といいます。また，横分割した表を**横分割表**といいます。なお，表を横分割する RD エリアは，それぞれ異なるディスクに配置することを原則とします。

### (1) 適用基準

次に示す場合に表を横分割することをお勧めします。

- データ量が多い場合
- 特定の時間帯にアクセスが集中する場合
- 表の分割単位でユーザ用 RD エリアの運用（表へのデータの格納，表の再編成，バックアップの取得など）をする場合

### (2) 定義方法

定義系 SQL の CREATE TABLE で定義します。定義例は「6.2　横分割表の作成」を参照してください。

## 12.3.2　表の横分割の種類

表を横分割する方法には，次に示す 2 種類があります。

- キーレンジ分割
- ハッシュ分割（フレキシブルハッシュ分割，FIX ハッシュ分割）

### (1) キーレンジ分割

**キーレンジ分割**とは，表を構成する列のうち，特定の列が持つ値の範囲を条件として表を横分割することです。なお，表を横分割するときの条件にした特定の列を**分割キー**といいます。表のデータがどの RD エリアに格納されているかどうかを意識したい場合に使用します。横分割の指定方法には，次に示す 2 種類があります。

#### (a) 格納条件指定

比較演算子を使用して，それぞれの RD エリアへの格納条件を指定します。一つの RD エリアに対して，格納条件で指定された一つの範囲だけを指定できます。

#### (b) 境界値指定

定数を使用して，それぞれの RD エリアに格納するデータの，境界となる値を指定します。一つの RD エリアに対して，境界値で区切られた複数の範囲を指定できます。なお，境界値指定の場合，マトリクス分割もできます。マトリクス分割については，「12.4　表のマトリクス分割」を参照してください。

### (2) ハッシュ分割

**ハッシュ分割**とは，表を構成する列が持つ値をハッシュ関数を使用して，均等に RD エリアに格納し，表を横分割することです。表を横分割するときに指定した特定の列を分割キーといいます。キーの範囲を意識しないで，表のデータを RD エリアに均等に格納したい場合に使用します。ハッシュ分割は，境界値指定のキーレンジ分割と組み合わせてマトリクス分割ができます。マトリクス分割については，「12.4　表の

マトリクス分割」を参照してください。

ハッシュ分割には**フレキシブルハッシュ分割**と FIX **ハッシュ分割**があります。

フレキシブルハッシュ分割では，表を分割して RD エリアに格納する場合，どの RD エリアに分割されるか定まりません。このため，検索処理では，該当する表があるすべてのバックエンドサーバが対象になります。

FIX ハッシュ分割では，表がどの RD エリアに分割されたかを HiRDB が認識します。このため，検索処理では，該当するデータがあると予測されるバックエンドサーバだけが対象になります。

### (a) 分割キーの選択方法

分割キーには次に示すようなキーを指定してください。

- キー値の偏りが少ない
- キーの値に重複が少ない

また，ハッシュ分割では，分割キーに単一列と複数列が選択できます。単一列を指定した場合，分割列のキー値の種類が少なかったり，キー値に偏りがあるとデータを均等に分割できないことがあります。この場合，分割する列名を複数指定して，データを RD エリアに均等に分割させるようにします。

### (b) ハッシュ関数の種類

ハッシュ分割で使用する**ハッシュ関数**の内容を次の表に示します。

**表 12-2 ハッシュ関数の内容**

| ハッシュ関数 | 説明 |
|---|---|
| HASH0 | このハッシュ関数は，分割に指定した列のデータ型が DATE，TIMESTAMP，CHAR(8)※1，又は CHAR(6)※1で，年月の値を使用してハッシングし，データを格納する RD エリアを月単位で循環させて割り当てる場合に使用します。分割キーは単一列を指定する必要があります。 |
| HASH1 | このハッシュ関数は，分割に指定した列のすべてのデータ型に使用できます。分割に指定したすべての列のデータの全バイト※2を使用してハッシュします。データ長が 0 バイト以上の列に指定できます。 |
| HASH2 | このハッシュ関数は，分割に指定した列のすべてのデータ型に使用できます。分割に指定したすべての列のデータの全バイト※2を使用してハッシュします。データ長が 0 バイト以上の列に指定できます。<br>HASH1 を指定してデータを均等に RD エリアに格納できなかった場合に指定します。 |
| HASH3 | このハッシュ関数は，分割に指定した列のデータ型が INTEGER，又は SMALLINT の場合に使用します。分割したすべての列の末尾 2 バイト※2を使用してハッシュします。データ長が 2 バイト以上の列に指定できます。 |
| HASH4 | このハッシュ関数は，分割に指定した列のデータ型が DATE の場合に使用します。分割に指定したすべての列の先頭 4 バイト※2を使用してハッシュします。データ長が 4 バイト以上の列に指定できます。 |
| HASH5 | このハッシュ関数は，分割に指定した列のデータ型が TIME の場合に使用します。分割に指定したすべての列の先頭 3 バイト※2を使用してハッシュします。データ長が 3 バイト以上の列に指定できます。 |
| HASH6 | このハッシュ関数は，分割に指定した列のすべてのデータ型に使用できます。DECIMAL の場合に最も適しています。分割に指定したすべての列の，データの全バイト※2を使用してハッシュします。データ長が 0 バイト以上の列に指定できます。 |
| HASHA | このハッシュ関数は，分割に指定した列のすべてのデータ型に使用できます。分割に指定したすべての列のデータの全バイト※2を使用してハッシュします。データ長が 0 バイト以上の列に指定できます。 |

| ハッシュ関数 | 説明 |
|---|---|
| HASHB | このハッシュ関数は，分割に指定した列のすべてのデータ型に使用できます。分割に指定したすべての列のデータの全バイト※2を使用してハッシュします。データ長が 0 バイト以上の列に指定できます。<br>HASHA を指定してデータを均等に RD エリアに格納できなかった場合に指定します。 |
| HASHC | このハッシュ関数は，分割に指定した列のデータ型が INTEGER，又は SMALLINT の場合に使用します。分割したすべての列の末尾 2 バイト※2を使用してハッシュします。データ長が 2 バイト以上の列に指定できます。 |
| HASHD | このハッシュ関数は，分割に指定した列のデータ型が DATE の場合に使用します。分割に指定したすべての列の先頭 4 バイト※2を使用してハッシュします。データ長が 4 バイト以上の列に指定できます。 |
| HASHE | このハッシュ関数は，分割に指定した列のデータ型が TIME の場合に使用します。分割に指定したすべての列の先頭 3 バイト※2を使用してハッシュします。データ長が 3 バイト以上の列に指定できます。 |
| HASHF | このハッシュ関数は，分割に指定した列のすべてのデータ型に使用できます。DECIMAL の場合に最も適しています。分割に指定したすべての列の，データの全バイト※2を使用してハッシュします。データ長が 0 バイト以上の列に指定できます。 |

注 1

- リバランス表でない場合は HASH0 ～ HASH6 のどれかを指定してください。HASH6 が最も均等にハッシングされるので，通常は HASH6 を指定してください。ただし，分割キーのデータによっては均等にならない場合もあるので，そのときにはほかのハッシュ関数を指定してください。
- リバランス表の場合は HASHA ～ HASHF のどれかを指定してください。HASHF が最も均等にハッシングされるので，通常は HASHF を指定してください。ただし，分割キーのデータによっては均等にならない場合もあるので，そのときにはほかのハッシュ関数を指定してください。

注 2

分割キーに LOB 列は指定できません。

注※ 1

データ型が CHAR(8) 及び CHAR(6) の場合の形式は次のようにしてください。

- CHAR(8) の場合：'YYYYMMDD'
- CHAR(6) の場合：'YYYYMM'

YYYY：0001 ～ 9999（年）
MM：01 ～ 12（月）
DD：01 ～該当年月の最終日（日）

注※ 2

VARCHAR 型，MVARCHAR 型，又は NVARCHAR 型は，後続の空白を無視してハッシュします。また，DECIMAL 型，INTERVAL YEAR TO DAY 型，又は INTERVAL HOUR TO SECOND 型で符号部が 'F' の場合は，'C' に変換してハッシュします。

### (c) ハッシュ関数の選択方法

#### 実際にデータベースにデータを格納して選択する方法

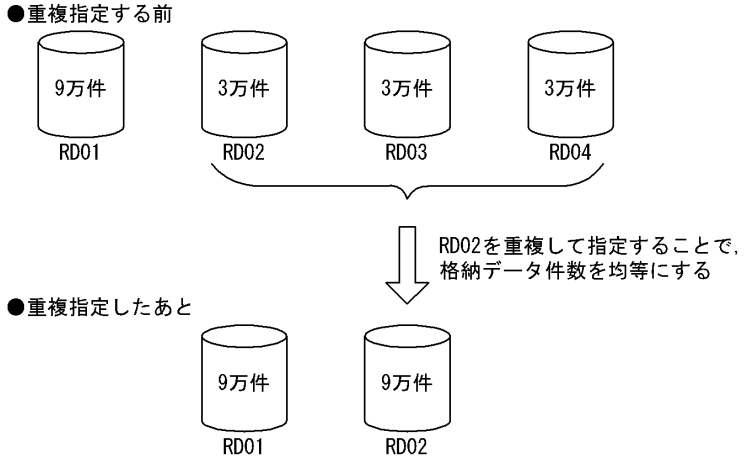
この場合のハッシュ関数の選択手順を次に示します。

1. 分割キーに対応して有効なハッシュ関数を指定します。
2. データベース状態解析ユティリティ（pddbst）で RD エリアごとに格納されている行数を確認します。
3. RD エリアごとに格納している行数に偏りがある場合には，ハッシュ関数を変更し，RD エリアご

との格納行数が均等になるようにします。

4. 3. の方法で格納行数が均等にならない場合は，格納行数の少ない RD エリアを重複して指定することで，格納行数が均等になるようにします。この場合の例を次の図に示します。

**図 12-4 ハッシュ分割で表格納用 RD エリアを重複指定する例**




**表分割ハッシュ関数を使用した UAP を作成し，ハッシュ関数を選択する方法**

この場合のハッシュ関数の選択手順を次に示します。

1. HiRDB からライブラリとして提供されている，表分割ハッシュ関数（分割キーのデータ値を入力すると分割条件指定順序を出力する関数）を使用して，RD エリアごとのデータ件数の偏りを求める UAP を作成します。
2. ハッシュ関数ごとに，表分割ハッシュ関数が出力する分割条件指定順序ごとの件数を求め，最も偏りが少ないハッシュ関数を選択します。

表分割ハッシュ関数を使用した UAP の作成方法については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

#### (d) ハッシュ関数が使用されるタイミング

ハッシュ関数は次に示すときに使用されます。

- 表単位のデータロード時
- データの追加時
- 表単位のデータのリロード時

### (3) キーレンジ分割，フレキシブルハッシュ分割及び FIX ハッシュ分割の相違点

キーレンジ分割，フレキシブルハッシュ分割及び FIX ハッシュ分割の相違点を次の表に示します。

**表 12-3 キーレンジ分割，フレキシブルハッシュ分割及び FIX ハッシュ分割の相違点**

| 相違点 | キーレンジ分割 | フレキシブル ハッシュ分割 | FIX ハッシュ分割 |
|---|---|---|---|
| データベース設計 | キーレンジを考慮して，データベースを設計する必要があります。 | キーレンジを考慮しないでデータベースを設計できます。 | キーレンジを考慮しないでデータベースを設計できます。 |

| 相違点 | キーレンジ分割 | フレキシブル<br>ハッシュ分割 | FIX ハッシュ分割 |
|---|---|---|---|
| 検索 | 検索条件によって，該当データが存在する可能性のあるバックエンドサーバだけ検索します。※1 | 対象表のある全バックエンドサーバで検索します。 | 探索条件に分割列に対する =,IN 述語を指定した場合に，該当データが存在する可能性のある RD エリアだけ検索します。ただし，ハッシュ関数に HASH0 を使用した場合は，= 述語 ,IN 述語 ,（<，>，<=，>= による）範囲条件 ,BETWEEN 述語 ,（前方一致比較の）LIKE 述語,（前方一致比較の）SIMILAR 述語を指定したときに検索します。※1 |
| データ増加時の対応 | キーが増加するデータの場合，データの格納が特定の RD エリアに偏ります。 | データが増加しても，常に均等に RD エリアに格納されます。 | データが増加しても，常に均等に RD エリアに格納されます。 |
| RD エリア閉塞時の運用 | 閉塞している RD エリアをアクセスしない検索条件であれば SQL を実行できます。※2 | 検索する表が格納されている RD エリアを一つでも閉塞すると，検索条件にかかわらず SQL を実行できません。 | 閉塞している RD エリアをアクセスしない検索条件であれば SQL を実行できます。※2 |
| 表の分割数の変更 | 表の再作成及び表の再編成が必要です。 | ALTER TABLE で RD エリアを追加でき，表の再編成は必須ではありません。 | 表の再作成及び表の再編成が必要です。ただし，表にデータが入っていないときだけ ALTER TABLE で RD エリアを追加できます。 |
| RD エリア単位のデータロード・リロード | 該当する RD エリアに格納するデータかどうかをチェックします。 | 該当する RD エリアに格納するデータかどうかをチェックしません。 | 該当する RD エリアに格納するデータかどうかをチェックします。 |
| データロード時の RD エリア単位による入力データファイルの作成方法 | キーレンジを考慮して，入力データを RD エリアごとに分類します。 | RD エリアごとのデータ件数が均等になるように，任意の方法で分類します。 | 表分割ハッシュ関数※3 を使用したアプリケーションを作成し，入力データを RD エリアごとに分類します。 |
| 分割キーの更新 | 同値更新だけできます。 | 更新できます。 | 同値更新だけできます。 |
| クラスタキーの UNIQUE 定義及び UNIQUE 指定のインデクス定義 | UNIQUE を指定できます。 | UNIQUE を指定できません。 | UNIQUE を指定できます。 |
| ALTER TABLE による分割格納条件の変更 | 次の分割方法の場合に変更できます。<br>• 境界値指定<br>• 格納条件指定（格納条件の比較演算子に = だけを使用している場合） | 変更できません。ただし，ALTER TABLE で RD エリアの追加はできます。 | 変更できません。ただし，ALTER TABLE で RD エリアの追加はできます。 |

注※ 1

ASSIGN LIST 文の場合，検索条件に該当しないバックエンドサーバにも負荷が掛かります。

注※ 2

ASSIGN LIST 文の場合，表全体が閉塞扱いになります。

注※ 3

表分割ハッシュ関数を使用した UAP の作成方法については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

## (4) 表の横分割定義時の指定規則

表の横分割定義時の指定規則を次に示します。

**● キーレンジ分割の場合**

- 指定できる分割キー※1は1個です。分割キーの更新はできません。
- 格納条件指定※2の場合，同じRDエリアを複数指定できません。境界値指定※3の場合，同じRDエリアを複数指定できますが，連続して同じRDエリアを指定することはできません。

**● ハッシュ分割の場合**

- 指定できる分割キー※1は最大16個です。ただし，同じ分割キーを重複して指定することはできません。フレキシブルハッシュ分割は，分割キーの更新ができますが，FIXハッシュ分割は分割キーの更新はできません。

注※1

次のデータ型の列及び繰返し列は，分割キーに指定できません。

- 定義長が256バイト以上のCHAR，VARCHAR，MCHAR，MVARCHAR型
- 定義長が28文字以上のNCHAR，NVARCHAR型
- BLOB型
- BINARY型
- 抽象データ型
- 小数秒精度が0より大きいTIMESTAMP型
- 既定値にCURRENT_TIMESTAMP USING BESを指定したTIMESTAMP型

注※2

格納条件を複数指定した場合，格納条件の指定順に条件を評価し，最初に真となった格納条件に指定したRDエリアに格納します。すべての条件で真とならない場合，格納条件を指定していないRDエリアに格納します。ただし，格納条件を指定していないRDエリアがない場合，どのRDエリアにも格納されません。また，条件を評価した結果，行が1行も格納されないRDエリアがある指定の表定義はできません。

注※3

境界値には定数を指定します。ただし，長さが0の文字列定数は指定できません。境界値を複数指定する場合，昇順となるように指定してください。また，境界値を指定しないRDエリアを最後に必ず指定してください。

## (5) キーレンジ分割（格納条件指定）の例

キーレンジ分割（格納条件指定）の例を次の図に示します。

**図 12-5　キーレンジ分割（格納条件指定）の例**

● USR01に格納された横分割表

ZAIKO（101L～353Mのデータ）

| SCODE | SNAME | COL | TANKA | ZSURYO |
|---|---|---|---|---|
| 101L | ブラウス | 青 | 3500 | 62 |
| 101M | ブラウス | 白 | 3500 | 85 |
| 201M | ポロシャツ | 白 | 3640 | 29 |
| 202M | ポロシャツ | 赤 | 3640 | 67 |
| 302S | スカート | 白 | 5110 | 65 |
| 353L | スカート | 赤 | 4760 | 18 |
| 353M | スカート | 緑 | 4760 | 56 |

● USR02に格納された横分割表

ZAIKO（411M～617Mのデータ）

| SCODE | SNAME | COL | TANKA | ZSURYO |
|---|---|---|---|---|
| 411M | セーター | 青 | 8400 | 12 |
| 412M | セーター | 赤 | 8400 | 22 |
| 591L | ソックス | 赤 | 250 | 300 |
| 591M | ソックス | 青 | 250 | 90 |
| 591S | ソックス | 白 | 250 | 280 |
| 671L | トレーナー | 白 | 4500 | 45 |
| 671M | トレーナー | 青 | 4500 | 76 |

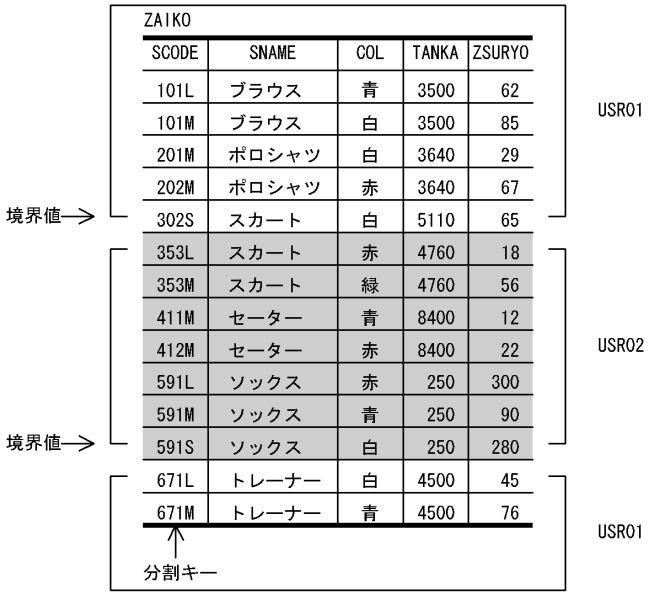
〔説明〕

ZAIKO 表の商品コード列（SCODE）の範囲（100L ～ 399S と 400L ～ 699S）を条件として，複数のユーザ用 RD エリア（USR01 と USR02）に横分割します。

## (6) キーレンジ分割（境界値指定）の例

キーレンジ分割（境界値指定）の例を次の図に示します。

**図 12-6 キーレンジ分割（境界値指定）の例**




● USR01に格納された横分割表

ZAIKO

| SCODE | SNAME | COL | TANKA | ZSURYO |
|---|---|---|---|---|
| 101L | ブラウス | 青 | 3500 | 62 |
| 101M | ブラウス | 白 | 3500 | 85 |
| 201M | ポロシャツ | 白 | 3640 | 29 |
| 202M | ポロシャツ | 赤 | 3640 | 67 |
| 302S | スカート | 白 | 5110 | 65 |
| 671L | トレーナー | 白 | 4500 | 45 |
| 671M | トレーナー | 青 | 4500 | 76 |

● USR02に格納された横分割表

ZAIKO

| SCODE | SNAME | COL | TANKA | ZSURYO |
|---|---|---|---|---|
| 353L | スカート | 赤 | 4760 | 18 |
| 353M | スカート | 緑 | 4760 | 56 |
| 411M | セーター | 青 | 8400 | 12 |
| 412M | セーター | 赤 | 8400 | 22 |
| 591L | ソックス | 赤 | 250 | 300 |
| 591M | ソックス | 青 | 250 | 90 |
| 591S | ソックス | 白 | 250 | 280 |

〔説明〕

ZAIKO 表の商品コード列（SCODE）の 302S，591S を境界値として，複数のユーザ用 RD エリア（USR01 と USR02）に横分割します。

## (7) フレキシブルハッシュ分割及び FIX ハッシュ分割の例

フレキシブルハッシュ分割，FIX ハッシュ分割の例を次の図に示します。

**図 12-7 フレキシブルハッシュ分割，FIX ハッシュ分割の例**

ZAIKO

| SCODE | SNAME | COL | TANKA | ZSURYO |
|---|---|---|---|---|
| 101L | ブラウス | 青 | 3500 | 62 |
| 101M | ブラウス | 白 | 3500 | 85 |
| 201M | ポロシャツ | 白 | 3640 | 29 |
| 202M | ポロシャツ | 赤 | 3640 | 67 |
| 302S | スカート | 白 | 5110 | 65 |
| 353L | スカート | 赤 | 4760 | 18 |
| 353M | スカート | 緑 | 4760 | 56 |
| 411M | セーター | 青 | 8400 | 12 |
| 412M | セーター | 赤 | 8400 | 22 |
| 591L | ソックス | 赤 | 250 | 300 |
| 591M | ソックス | 青 | 250 | 90 |
| 591S | ソックス | 白 | 250 | 280 |
| 671L | トレーナー | 白 | 4500 | 45 |
| 671M | トレーナー | 青 | 4500 | 76 |

↑
分割キー

● USR01に格納された横分割表

ZAIKO

| SCODE | SNAME | COL | TANKA | ZSURYO |
|---|---|---|---|---|
| 101L | ブラウス | 青 | 3500 | 62 |
| 353M | スカート | 緑 | 4760 | 56 |
| 411M | セーター | 青 | 8400 | 12 |
| 412M | セーター | 赤 | 8400 | 22 |
| 591M | ソックス | 青 | 250 | 90 |
| 591S | ソックス | 白 | 250 | 280 |
| 671M | トレーナー | 青 | 4500 | 76 |

● USR02に格納された横分割表

ZAIKO

| SCODE | SNAME | COL | TANKA | ZSURYO |
|---|---|---|---|---|
| 101M | ブラウス | 白 | 3500 | 85 |
| 201M | ポロシャツ | 白 | 3640 | 29 |
| 202M | ポロシャツ | 赤 | 3640 | 67 |
| 302S | スカート | 白 | 5110 | 65 |
| 353L | スカート | 赤 | 4760 | 18 |
| 591L | ソックス | 赤 | 250 | 300 |
| 671L | トレーナー | 白 | 4500 | 45 |

〔説明〕

ZAIKO 表の商品コード列（SCODE）を分割キーとして，ハッシュ関数 HASH6 を使用して複数のユーザ用 RD エリア（USR01 と USR02）に横分割します。

なお，実際のデータの格納先 RD エリアはこの例と異なることがあります。

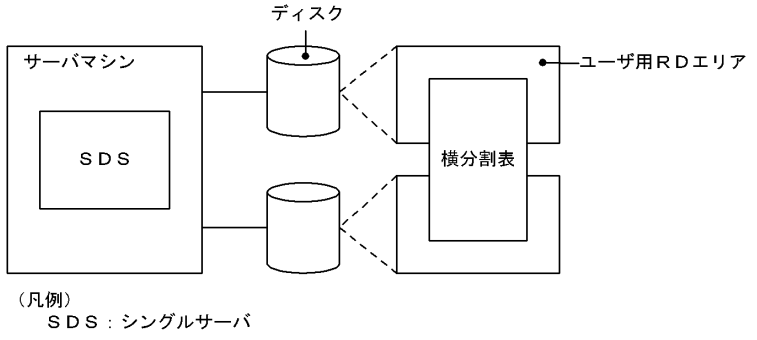
## 12.3.3 表の横分割の形態

表の横分割の基本的な形態には，次に示す 2 種類があります。

- サーバ内の横分割（HiRDB/ シングルサーバでの形態）
- サーバ間の横分割（HiRDB/ パラレルサーバでの形態）

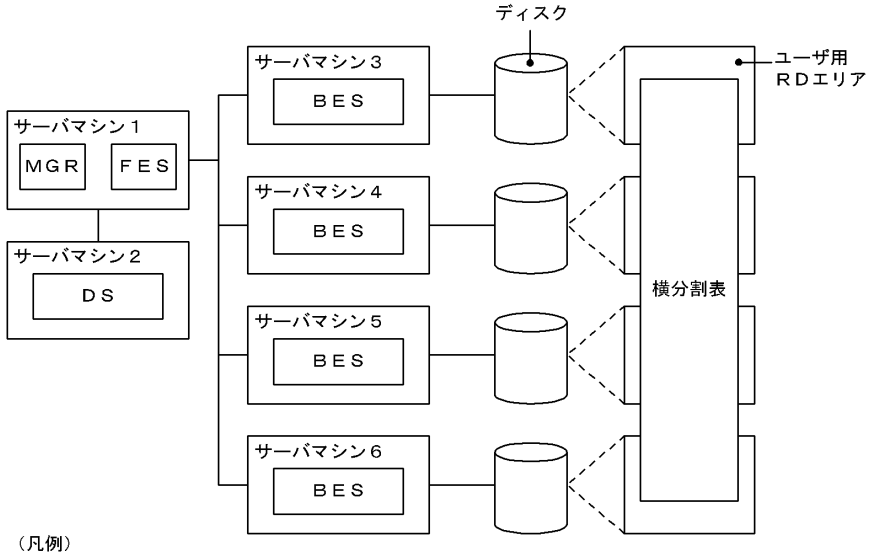
それぞれの形態を図 12-8 及び図 12-9 に示します。

**図 12-8 表の横分割の形態（HiRDB/ シングルサーバの場合）**




（凡例）
ＳＤＳ：シングルサーバ

**図 12-9 表の横分割の形態（HiRDB/ パラレルサーバの場合）**




（凡例）
ＭＧＲ：システムマネジャ
ＦＥＳ：フロントエンドサーバ
ＤＳ　：ディクショナリサーバ
ＢＥＳ：バックエンドサーバ

## 12.3.4 表の横分割の効果

表を横分割して得られる効果を次に示します。

### (1) HiRDB/ シングルサーバの場合

**操作性の向上**

ユーザ用 RD エリアごとに，表へのデータの格納，表の再編成，バックアップの取得などの運用ができます。

**キーレンジ分割の場合**

ディクショナリ表の SQL_DIV_TABLE 表を検索することで，表のデータをどのユーザ用 RD エリア

に格納したかが分かります。このため，ユーザ用 RD エリアに障害が発生した場合に，どのデータが利用できないかが分かります。なお，ディクショナリ表の検索方法と SQL_DIV_TABLE 表については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

### (2) HiRDB/ パラレルサーバの場合

**性能の向上**

- 表にアクセスする処理を複数のユーザ用 RD エリアにわたって並列化できるため，表に対するアクセスの高速化が図れます。
- 表にアクセスする処理の負荷を複数のバックエンドサーバに分散できます。

**操作性の向上**

効果は HiRDB/ シングルサーバの場合と同様です。

## 12.3.5 設計上の考慮点

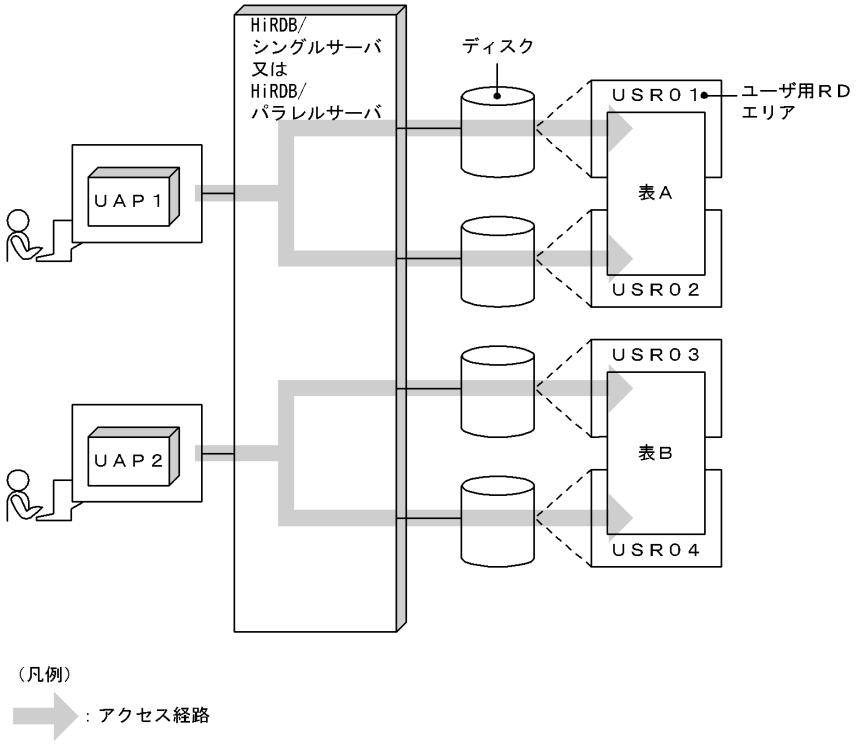
### (1) HiRDB/ シングルサーバと HiRDB/ パラレルサーバでの共通の考慮点

HiRDB/ シングルサーバと HiRDB/ パラレルサーバでの共通の考慮点を次に示します。

#### (a) ディスクに対するアクセスの競合を考慮した横分割

複数の UAP がそれぞれ別々の表に同時にアクセスする場合は，これらの表をそれぞれ異なるディスク上の異なるユーザ用 RD エリアにわたって横分割します。ディスクに対するアクセスの競合を考慮した横分割の概要を次の図に示します。

**図 12-10 ディスクに対するアクセスの競合を考慮した横分割の概要**




〔説明〕

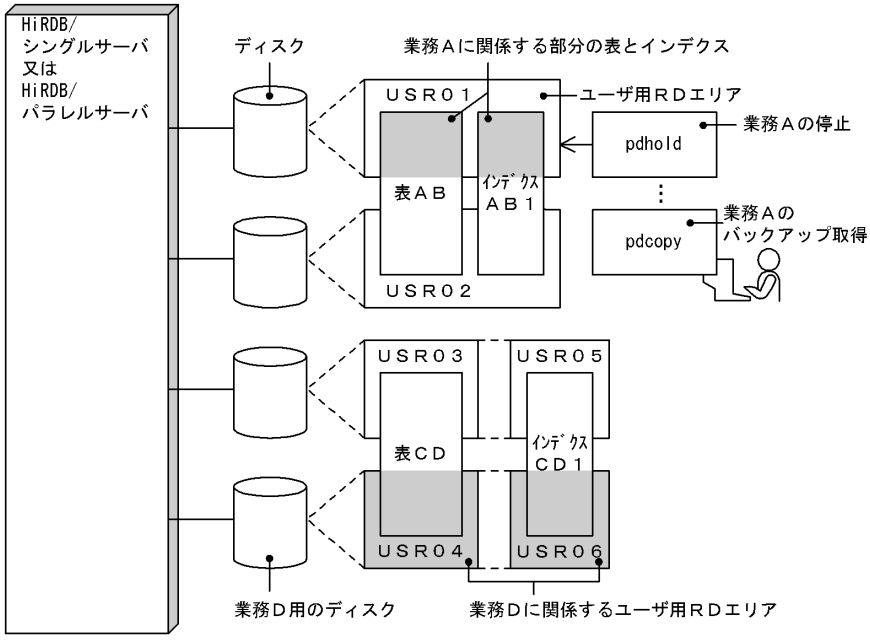
表 A と表 B をそれぞれ異なるディスク上のユーザ用 RD エリア USR01 ～ USR02 と USR03 ～ USR04 にわたって横分割しています。このため，UAP1 と UAP2 が同時に表 A と表 B にアクセスしても，ディスクに対するアクセスの競合による待ちが発生しないため，待ち時間を短くできます。

しかし，一つのディスク上のユーザ用 RD エリアに複数の表を格納した場合，これらの表に対して複数の UAP が同時にアクセスすると，ディスクに対するアクセスの競合が発生します。このため，この表にアクセスできた一つの UAP のアクセスが終了するまで，ほかの UAP が待たされることになり，待ち時間が長くなります。

### (b) 操作性を考慮した横分割

操作性を考慮した横分割の概要を次の図を基に説明します。

**図 12-11　操作性を考慮した横分割の概要**




〔説明〕

- **表とインデクスの同一ユーザ用 RD エリアへの格納**
  検索性能よりはむしろ，表の作成，表の再編成，ユーザ用 RD エリアのバックアップの取得，RD エリアの回復などの運用の操作性を重視する場合は，横分割した表とそれに対応するインデクスを同じユーザ用 RD エリアに格納します。これによって，ユーザ用 RD エリアごとの独立した運用が便利になります。
  図 12-11 の例では，表 AB のうち，業務 A に関係する部分の表とインデクスを専用のユーザ用 RD エリア USR01 にまとめて格納しています。これによって，例えば，業務 A を停止する場合は，pdhold コマンド（RD エリアの閉塞）で運用できます。また，データベース複写ユティリティ（pdcopy）を使用した業務単位でのバックアップが取得しやすくなります。
- **関連するユーザ用 RD エリアの同一ディスクへの配置**
  横分割した表とそれに対応するインデクスをそれぞれ異なるユーザ用 RD エリアに格納する場合は，これらの互いに関連するユーザ用 RD エリアを同一のディスクに配置します。これによって，ディスクごとに独立したユーザ用 RD エリアの運用ができます。
  図 12-11 の例では，表 CD のうち，業務 D に関係する部分の表とインデクスをそれぞれ格納したユーザ用 RD エリア USR04 と USR06 を同一のディスクに配置しています。これによって，ディスクごとに業務の運用ができます。

## (2) HiRDB/ パラレルサーバ固有の考慮点

HiRDB/ パラレルサーバ固有の考慮点を次に示します。

### (a) ディスクに対するアクセスの負荷を考慮した横分割

- **複数のバックエンドサーバにわたる横分割**
  一つのバックエンドサーバのディスクに複数のユーザ用 RD エリアを配置したとき，それぞれのユーザ用 RD エリアに格納された表のアクセス頻度がどれも高い場合，このバックエンドサーバでのディスク

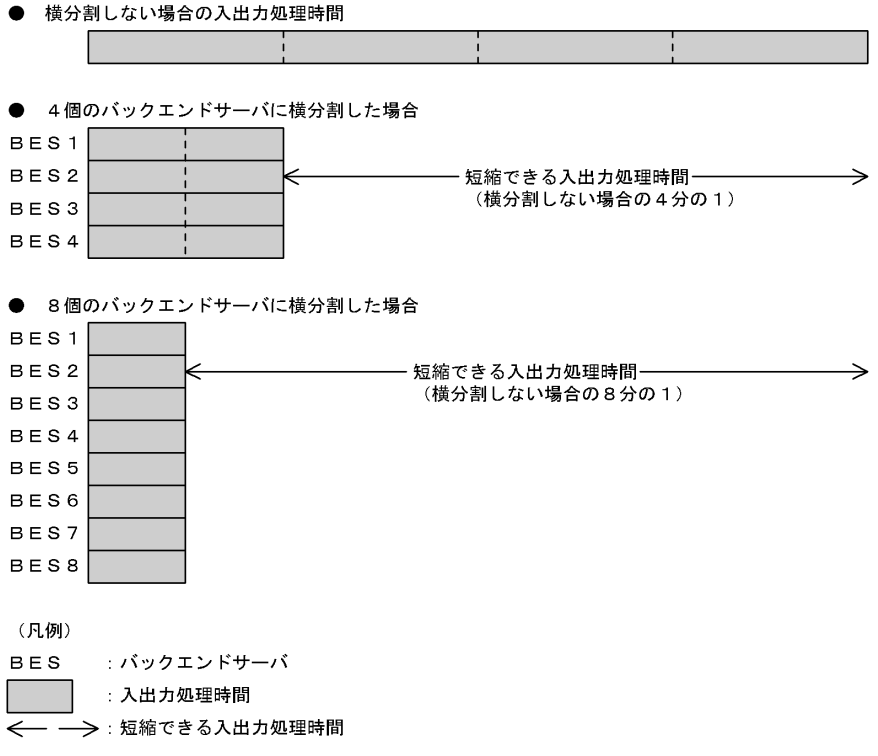
に対するアクセスの負荷が高くなります。
このため，アクセス頻度の高い表は，複数のバックエンドサーバの異なるディスク上のユーザ用 RD エリアにわたって横分割します。このとき，それぞれのバックエンドサーバでの表に対するアクセス頻度が均等になるようにします。

- **複数のサーバマシンにわたるディスクアクセスの並列化**
  あるサーバマシンの一つのバックエンドサーバのユーザ用 RD エリアに格納された表に対する処理が，入出力が主体で CPU の負荷が低い場合，ディスクに対するアクセスの負荷が複数のサーバマシンにわたって均等にならないため，並列処理の効率が下がります。
  このため，サーバマシン内の CPU ビジー率に余裕がある場合は，このサーバマシンに更にバックエンドサーバとユーザ用 RD エリアを配置して，ディスクに対するアクセスの並列度を向上させるようにします。

### (b) 入出力処理の並列度を考慮した横分割

表をできるだけ多くのサーバマシンにわたって横分割すると，並列処理によって表に対する入出力処理時間が短縮できます。横分割できるサーバマシン数に限界があれば，各サーバマシンにバックエンドサーバとディスクを増やして表を横分割することで，入出力処理の並列効果が得られます。表を横分割するバックエンドサーバの数に伴う入出力処理性能の概要を次の図に示します。

**図 12-12 表を横分割するバックエンドサーバの数に伴う入出力処理性能の概要**




ただし，表を横分割するバックエンドサーバの数が多過ぎると，各バックエンドサーバでの処理結果をフロントエンドサーバに返すための通信量が増加します。このため，データベースの運用や SQL 処理（容量の大きい表から大量のデータを検索する SQL 処理かどうか）を考慮して，表を横分割するバックエンドサーバの数を決定する必要があります。

### (c) 表に対するアクセス頻度を考慮した横分割

それぞれのバックエンドサーバでの表に対するアクセス頻度が均等になるように表を横分割します。

アクセス頻度を均等にするための考慮点を次に示します。

**キーレンジ分割の場合**

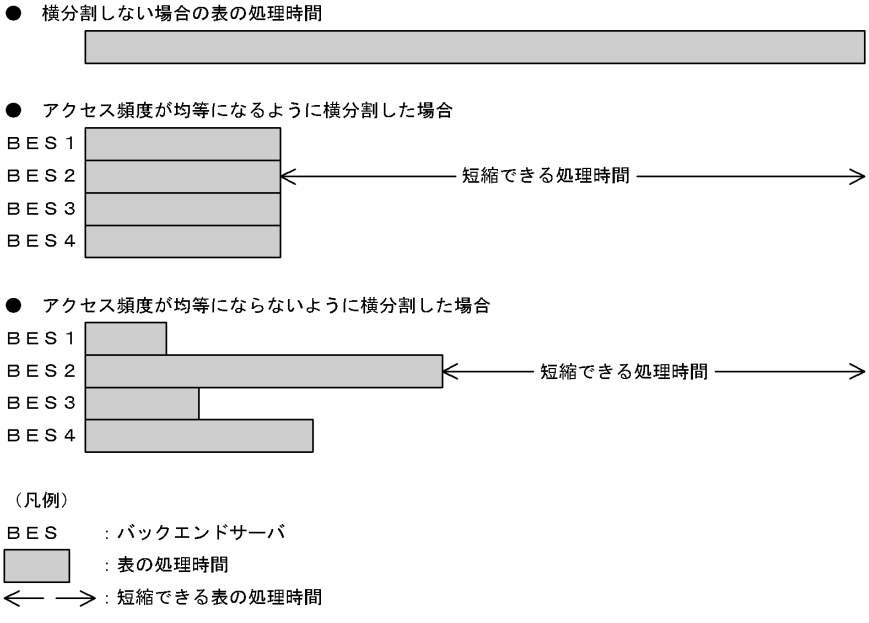
- 表の横分割を定義する場合に，分割キーに UNIQUE を指定して，データ量が均等になるようにします。
- 表を横分割する場合に，あるキーレンジのデータに対するアクセス回数が，ほかのキーレンジのデータに対するアクセス回数よりも多いと予想できるときは，アクセス回数が多いと予想できるキーレンジのデータを更にキーレンジで分割します。

**フレキシブルハッシュ分割，FIX ハッシュ分割の場合**

- ハッシュ関数を変更して，データ量が均等になるように調整します。
- 分割キーに偏りや重複がないものを選択して，データ量が均等になるように調整します。

複数のバックエンドサーバにわたって表を横分割する場合でも，表に対するアクセス頻度が均等になるように横分割した場合と，均等にならないように横分割した場合とでは，表に対する並列処理の性能に差が生じます。表に対するアクセス頻度の違いによる並列処理性能の違いを次の図に示します。

**図 12-13　表に対するアクセス頻度の違いに伴う並列処理性能の違い**




〔説明〕

アクセス頻度が均等になるように表を横分割した場合と，均等にならないように表を横分割した場合とでは，削減できる処理時間が異なります。均等にならないように表を横分割した場合は，バックエンドサーバ BES2 での処理が終了するまで，表に対する処理は終了しないため，並列処理の効果が得られません。

### (d) 複雑な検索処理を考慮した横分割

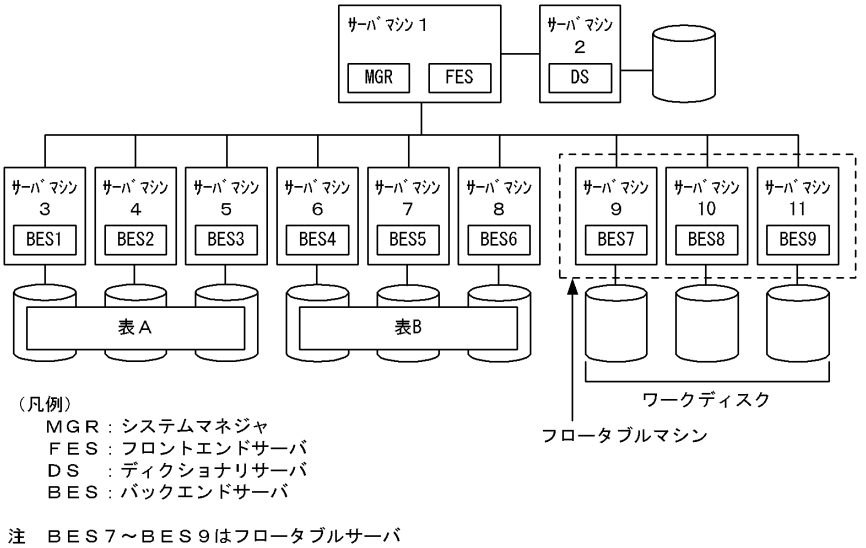
大量のデータの検索や結合処理をするなど，複雑な検索処理を考慮した表の横分割をする場合，次に示す

手順で設計します。

1. **ディスクに対する処理時間とディスクの台数の決定**
   データの規模と処理のパターンから，ディスクへのアクセス頻度（利用率）を求め，これを基にデータをディスクに配分して，ディスクに対する処理時間の目標値を決めます。なお，結合処理をする場合は，結合処理に必要なワークディスク（ソートマージ用に使用する作業表用ファイルを作成するHiRDB ファイルシステム領域の数）を除いてデータを配分します。結合処理に必要な時間は，ディスクに対する処理時間の目標値から除きます。ディスクへのデータ配分から，ディスクの台数を決定します。
2. **サーバマシンの台数の決定**
   データの処理パターンから，サーバマシンでの処理のオーバヘッド時間を求めます。ディスクに対する処理時間とサーバマシンでのオーバヘッド時間が，それぞれ均等になるようにサーバマシンの台数（バックエンドサーバを配置するサーバマシンの台数）を決定します。
3. **結合処理をするサーバマシンの台数の決定**
   データの処理パターンから，結合処理をするときに掛かるサーバマシンでのオーバヘッド時間を求めます。この時間とディスクに対する処理時間を基に，必要とするフロータブルマシンの台数を決定します。
   **フロータブルマシン**とは，結合処理のような複雑な検索処理をするための専用のバックエンドサーバ（**フロータブルサーバ**）を配置するサーバマシンのことです。フロータブルサーバとするバックエンドサーバは，ユーザ用 RD エリアを割り当てないバックエンドサーバとして定義します。
4. **ワークディスクの台数の決定**
   フロータブルマシンには，結合処理の対象となるデータが各バックエンドサーバから均等に分配されます。このときに予想されるデータ量を基に，ワークディスク（作業表用ファイルを作成する HiRDB ファイルシステム領域）の台数を求めます。
5. **システム構成の決定**
   以上で決定したサーバマシンとディスクの台数から，システム全体の構成を決定します。

以上の手順で設計した，複雑な検索処理を考慮した横分割のシステム構成の概念を次の図に示します。

**図 12-14　複雑な検索処理を考慮した横分割のシステム構成**




注　ＢＥＳ７～ＢＥＳ９はフロータブルサーバ

〔説明〕

バックエンドサーバ BES1 ～ BES3 とバックエンドサーバ BES4 ～ BES6 が，それぞれ表 A と表 B から結合処理の対象となるデータを読み出します。フロータブルサーバ BES7 ～ BES9 は，バックエンドサーバ BES1 ～ BES6 が読み出したデータを受け取り，並列に突き合わせる処理をします。
このようなシステム構成にすることで，バックエンドサーバ BES1 ～ BES6 での負荷を軽減でき，処理時間を短縮できます。なお，フロータブルサーバを配置しなかった場合は，フロータブルサーバで実行していた結合処理をバックエンドサーバ BES1 ～ BES6 のどれかが実行することになります。

### 12.3.6　表を横分割する場合の注意

1. 表を横分割した場合は，この表に作成するインデクスも横分割してください。インデクスを一つのユーザ用 RD エリアに格納して，表を横分割して運用すると，インデクスでの排他制御によって，同時実行性が低下することがあります。インデクスの横分割については，「13.3　インデクスの横分割」を参照してください。
2. 次に示す場合は，**ハッシュ分割表のリバランス機能**を使用することをお勧めします。
   - 表をハッシュ分割する場合
   - データ量の増加が見込まれる場合

   ハッシュ分割表のデータ量が増加したため RD エリアを追加すると（表の横分割数を増やすと），既存の RD エリアと新規追加した RD エリアとの間でデータ量の偏りが生じます。ハッシュ分割表のリバランス機能を使用すると，表の横分割数を増やすときにデータ量の偏りを修正できます。ハッシュ分割表のリバランス機能については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

# 12.4　表のマトリクス分割

表の二つの列を分割キーとして，分割方法の指定を組み合わせて分割することを**マトリクス分割**といいます。一つ目の分割キーとなる列を**第 1 次元分割列**，二つ目の分割キーとなる列を**第 2 次元分割列**といいます。マトリクス分割は，第 1 次元分割列で境界値指定のキーレンジ分割をし，分割されたデータをさらに第 2 次元分割列で分割します。第 2 次元分割列に指定できる分割方法を次に示します。

- 境界値指定のキーレンジ分割
- フレキシブルハッシュ分割※
- FIX ハッシュ分割※

注※
　指定できるハッシュ関数は HASH0 ～ HASH6 です。HASHA ～ HASHF は指定できません。

マトリクス分割によって分割された表を**マトリクス分割表**といいます。

なお，表をマトリクス分割するためには，HiRDB Advanced Partitioning Option が必要です。

## (1) 表のマトリクス分割の効果

複数の列を分割キーとして分割することで得られる効果を次に示します。

- SQL 処理の高速化
  SQL の処理を並列に実行したり，複数のキーによる検索で検索範囲を絞り込んで高速に処理したりできます。
- 運用時間の短縮
  より細かな分割ができるため，1RD エリアの大きさを小さくして，再編成，バックアップの取得，障害発生時の回復作業などに要する時間を短縮できます。

## (2) 適用基準

次の場合，境界値指定のキーレンジ分割の組み合わせをお勧めします。

- 第 1 次元分割列による分割では，各境界値に該当するデータ量が多大となる
- 表にアクセスする UAP で指定できる探索条件に指定する列が複数あり，複数の列でアクセスする RD エリアを限定したい，又は一つの SQL 文中で n 番目に指定した列だけでアクセスする RD エリアを限定したい

次の場合，境界値指定のキーレンジ分割とハッシュ分割の組み合わせをお勧めします。

- 第 1 次元分割列による分割では，各境界値に該当するデータ量が多大となる
- 第 1 次元分割列で分割された範囲のデータを，均等に細分化して格納したい

次の場合，境界値指定のキーレンジ分割とハッシュ分割の組み合わせで，ハッシュ分割に指定する RD エリア名を重複指定することをお勧めします。同じ RD エリア名を重複指定することによって，分割数はそのままで，実際に使用する RD エリア数を減らすことができます。また，各 RD エリアに格納するデータ量も均等にできます。RD エリア名を重複指定する場合については，「図 12-4　ハッシュ分割で表格納用 RD エリアを重複指定する例」を参照してください。

- 1RD エリアのサイズを一定にすることで各 RD エリアの再編成処理の時間を一定にしたいが，第 1 次元分割列の境界値による分割ではデータ量を均等に分ける境界値の指定が難しいため，一定にできない
- 分割数が検索時の性能に影響しているため，分割数を減らしたい

### (3) 定義方法

定義系 SQL の CREATE TABLE の PARTITIONED BY MULTIDIM オペランドで次の指定をします。

- RD エリアへの表の割り当て
- マトリクス分割の方法（分割キー，分割方法）

定義時の規則を次に示します。

**● 第 2 次元分割列が境界値指定のキーレンジ分割の場合**

- 分割キーの指定可能数は 2（第 1 次元分割列の分割キー＋第 2 次元分割列の分割キー）です。第 1 次元分割列と第 2 次元分割列で同じ分割キーは指定できません。

**● 第 2 次元分割列がハッシュ分割の場合**

- 分割キーの指定可能数は 2 ～ 16 です。フレキシブルハッシュ分割のとき，第 2 次元分割列の分割キーだけは更新できます。FIX ハッシュ分割のとき，分割キーの更新はできません。

定義例は「(4) マトリクス分割の例」を参照してください。

### (4) マトリクス分割の例

#### (a) 境界値指定のキーレンジ分割の組み合わせの場合

顧客表の登録日及び店番号に境界値を指定し，登録日と店番号によって，表をマトリクス分割します。それぞれの顧客データを次のようにユーザ用 RD エリア（USR01 ～ USR06）に格納します。格納するのに必要なユーザ用 RD エリア数は，（境界値数＋ 1）×（境界値数＋ 1）なので，この例の場合，3 × 2 ＝ 6 です。

| 登録日 | 店番号 | |
|---|---|---|
| | 100 以下 | 100 より大きい |
| 2000 年以前 | USR01 | USR02 |
| 2001 年 | USR03 | USR04 |
| 2002 年以降 | USR05 | USR06 |

マトリクス分割する表を定義する SQL 文を次に示します。

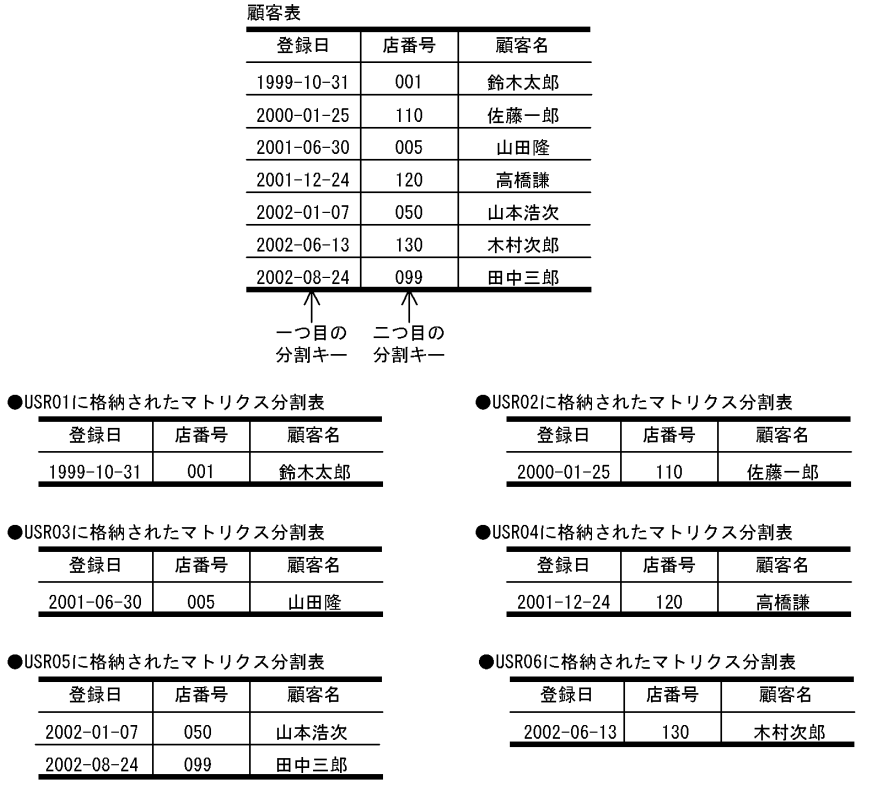
```
CREATE FIX TABLE 顧客表
  (登録日 DATE, 店番号 INT, 顧客名 NCHAR(10))
  PARTITIONED BY MULTIDIM(
  登録日 (('2000-12-31'),('2001-12-31')), …1.
  店番号 ((100))                          …2.
  )IN ((USR01,USR02),(USR03,USR04),(USR05,USR06))
```

〔説明〕

1. 第 1 次元分割列名（一つ目の分割キーとなる列名）と，その境界値リストを指定します。
2. 第 2 次元分割列名（二つ目の分割キーとなる列名）と，その境界値リストを指定します。

マトリクス分割の例を次の図に示します。

**図 12-15 マトリクス分割の例（境界値指定のキーレンジ分割の組み合わせ）**




### (b) 境界値指定のキーレンジ分割とハッシュ分割の組み合わせの場合

第 2 次元分割列で FIX ハッシュ分割する場合の例について説明します。

顧客表の登録日に境界値を指定し，店番号と地域コードをハッシュ関数で三分割することによって，表をマトリクス分割します。それぞれの顧客データを次のようにユーザ用 RD エリア（USR01 ～ USR09）に格納します。格納するのに必要なユーザ用 RD エリア数は，（境界値数＋ 1）×（ハッシュ関数で分割するユーザ任意の数）なので，この例の場合，3 × 3 ＝ 9 です。

| 登録日 | 店番号と地域コード（ハッシュ関数で三分割） | | |
|---|---|---|---|
| 2002 年以前 | USR01 | USR02 | USR03 |
| 2003 年 | USR04 | USR05 | USR06 |
| 2004 年以降 | USR07 | USR08 | USR09 |

マトリクス分割する表を定義する SQL 文を次に示します。

```
CREATE FIX TABLE 顧客表
  (登録日 DATE, 店番号 INT, 地域コード INT, 顧客名 NCHAR(10))
  PARTITIONED BY MULTIDIM
  (登録日 (('2002-12-31'),('2003-12-31')), …1.
  FIX HASH HASH6 BY 店番号, 地域コード      …2.
  )IN ((USR01,USR02,USR03),
       (USR04,USR05,USR06),
```

```
        (USR07,USR08,USR09))
```

〔説明〕

1. 第 1 次元分割列名（一つ目の分割キーとなる列名）と，その境界値リストを指定します。
2. 第 2 次元列分割名（二つ目の分割キーとなる列名）と，ハッシュ関数名を指定します。

マトリクス分割の例を次の図に示します。

**図 12-16　マトリクス分割の例（境界値指定のキーレンジ分割とハッシュ分割の組み合わせ）**

顧客表

| 登録日 | 店番号 | 地域コード | 顧客名 |
|---|---|---|---|
| 2002-01-31 | 001 | 01 | 鈴木太郎 |
| 2002-05-25 | 110 | 03 | 佐藤一郎 |
| 2002-06-30 | 005 | 01 | 山田隆 |
| 2003-01-24 | 120 | 03 | 高橋謙 |
| 2003-03-07 | 050 | 01 | 山本浩次 |
| 2003-04-13 | 130 | 03 | 木村次郎 |
| 2003-08-24 | 099 | 02 | 田中三郎 |
| 2003-11-15 | 010 | 01 | 山口美佐 |
| 2003-12-24 | 020 | 01 | 大田五郎 |
| 2004-01-27 | 080 | 02 | 斎藤和美 |
| 2004-03-24 | 170 | 04 | 木下美智子 |
| 2004-07-24 | 015 | 01 | 大川靖男 |

↑ 一つ目の分割キー　↑ 二つ目の分割キー　↑ 二つ目の分割キー

●USR01に格納されたマトリクス分割表

| 登録日 | 店番号 | 地域コード | 顧客名 |
|---|---|---|---|
| 2002-01-31 | 001 | 01 | 鈴木太郎 |

●USR02に格納されたマトリクス分割表

| 登録日 | 店番号 | 地域コード | 顧客名 |
|---|---|---|---|
| 2002-05-25 | 110 | 03 | 佐藤一郎 |

●USR03に格納されたマトリクス分割表

| 登録日 | 店番号 | 地域コード | 顧客名 |
|---|---|---|---|
| 2002-06-30 | 005 | 01 | 山田隆 |

●USR04に格納されたマトリクス分割表

| 登録日 | 店番号 | 地域コード | 顧客名 |
|---|---|---|---|
| 2003-01-24 | 120 | 03 | 高橋謙 |
| 2003-08-24 | 099 | 02 | 田中三郎 |

●USR05に格納されたマトリクス分割表

| 登録日 | 店番号 | 地域コード | 顧客名 |
|---|---|---|---|
| 2003-03-07 | 050 | 01 | 山本浩次 |
| 2003-11-15 | 010 | 01 | 山口美佐 |

●USR06に格納されたマトリクス分割表

| 登録日 | 店番号 | 地域コード | 顧客名 |
|---|---|---|---|
| 2003-04-13 | 130 | 03 | 木村次郎 |
| 2003-12-24 | 020 | 01 | 大田五郎 |

●USR07に格納されたマトリクス分割表

| 登録日 | 店番号 | 地域コード | 顧客名 |
|---|---|---|---|
| 2004-01-27 | 080 | 02 | 斎藤和美 |

●USR08に格納されたマトリクス分割表

| 登録日 | 店番号 | 地域コード | 顧客名 |
|---|---|---|---|
| 2004-03-24 | 170 | 04 | 木下美智子 |

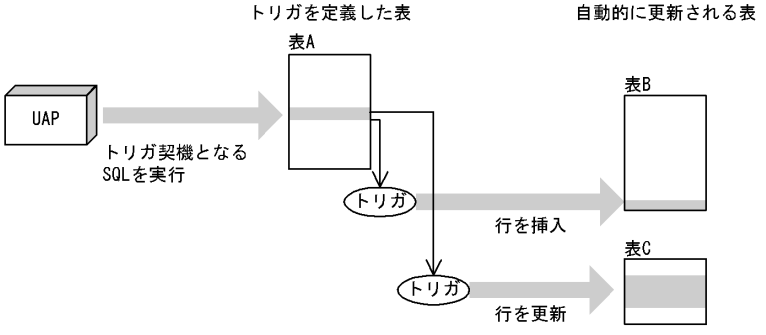
●USR09に格納されたマトリクス分割表

| 登録日 | 店番号 | 地域コード | 顧客名 |
|---|---|---|---|
| 2004-07-24 | 015 | 01 | 大川靖男 |

# 12.5　トリガの定義

トリガを定義すると，ある表への操作（更新，挿入，及び削除）を契機に自動的に SQL 文を実行させることができます。トリガは，定義する表，トリガを動作させる契機となる SQL（**トリガ契機となる SQL**），自動的に実行させる SQL 文（**トリガ SQL 文**），その動作が実行される条件（**トリガ動作の探索条件**）などを指定して定義します。トリガを定義した表にトリガ動作の探索条件を満たす SQL 文が実行されると，トリガ SQL 文が自動的に実行されます。トリガの概要を次の図に示します。

**図 12-17　トリガの概要**




〔説明〕

UAP からトリガ契機となる SQL が実行されると，トリガを定義した表 A でトリガが呼び出され，トリガ動作の探索条件を満たしている場合，自動的にトリガ SQL 文（この場合，表 B への行の挿入や表 C への行の更新）が実行されます。

**前提条件**

トリガを定義する場合，データディクショナリ LOB 用 RD エリアを作成しておく必要があります。データディクショナリ LOB 用 RD エリアは，データベース構成変更ユティリティ（pdmod）で作成します。

なお，トリガを表に定義すると，その表を使用する関数，手続き及びトリガの SQL オブジェクトは無効になるため，SQL オブジェクトを再作成する必要があります。また，トリガが使用しているリソース（表，インデクスなど）が定義，定義変更，又は削除された場合，トリガの SQL オブジェクトは無効になるため，SQL オブジェクトを再作成する必要があります。詳細は「12.5.4　トリガの管理」を参照してください。

## 12.5.1　適用基準

UAP で次のような処理をする場合，トリガの使用をお勧めします。

- ある表の更新に伴ってほかの表を必ず更新する
- ある表の更新に伴って，その更新行中のある列を必ず更新する（列と列を関連づける）

例えば，商品管理表の価格が変更されると商品管理履歴表に変更内容を蓄積するという場合，トリガを使用しないと，商品管理表を更新する UAP は常に商品管理履歴表も更新する必要があります。トリガを使用すると，商品管理履歴表への操作を自動化できるため，商品管理表を更新する UAP は商品管理履歴表の更新を考慮する必要がありません。このように，トリガを適切に使用すると UAP を作成するときの負荷を軽減できます。

## 12.5.2 トリガの定義

### (1) 定義の準備

トリガを定義すると，指定したトリガ SQL 文を基にトリガ動作手続きの SQL オブジェクトが自動的に作成され，データディクショナリ LOB 用 RD エリアに格納されます。そのため，トリガを定義する場合，データディクショナリ LOB 用 RD エリアに十分な容量を確保しておく必要があります。データディクショナリ LOB 用 RD エリアの容量の見積もりについては，「16.5　データディクショナリ LOB 用 RD エリアの容量の見積もり」を参照してください。

また，トリガ契機となる SQL を実行するためには，SQL オブジェクト用バッファ長を指定するときにトリガの SQL オブジェクトについても考慮しておく必要があります。SQL オブジェクト用バッファ長の見積もりについては，マニュアル「HiRDB Version 8 システム定義」を参照してください。

### (2) 定義方法

トリガの定義，SQL オブジェクトの再作成，及び削除には次の定義系 SQL を使用します。

● CREATE TRIGGER

トリガを定義します。トリガは所有する表にだけ定義でき，他ユーザが所有する表には定義できません。次の項目を指定します。

- トリガ動作の実行時期
  トリガ動作は表への操作の前（BEFORE），又は後（AFTER）に実行できます。なお，トリガ動作時期に BEFORE を指定したトリガを BEFORE **トリガ**，AFTER を指定したトリガを AFTER **トリガ**といいます。
- トリガ動作の実行契機
  トリガ動作を実行する契機には，INSERT 文，DELETE 文，又は UPDATE 文の実行があります。
- トリガを定義する表
  トリガは実表にだけ定義できます。
- トリガ契機となる SQL の実行前後の行の名称（新旧値別名）
  トリガ契機となる SQL で更新する行に，SQL 実行前の名称（旧値相関名），又は実行後の名称（新値相関名）を指定します。ここで指定した名称を使ってトリガの動作内容を指定できます。
- トリガ動作
  トリガ動作には次の三つの要素があります。
  ・トリガ SQL 文（自動的に実行させる SQL 文）
  ・トリガ動作の探索条件（トリガ SQL 文が実行される条件）
  ・動作が実行されるのが行単位か文単位かの種別
  トリガ SQL 文はトリガ動作の探索条件を満たす場合にだけ実行され，条件を省略した場合はトリガ契機となる SQL が実行されると常にトリガ SQL 文が実行されます。

● ALTER TRIGGER

既に定義されているトリガの SQL オブジェクトを再作成します。定義系 SQL の ALTER ROUTINE でも再作成できます。

● DROP TRIGGER

トリガを削除します。

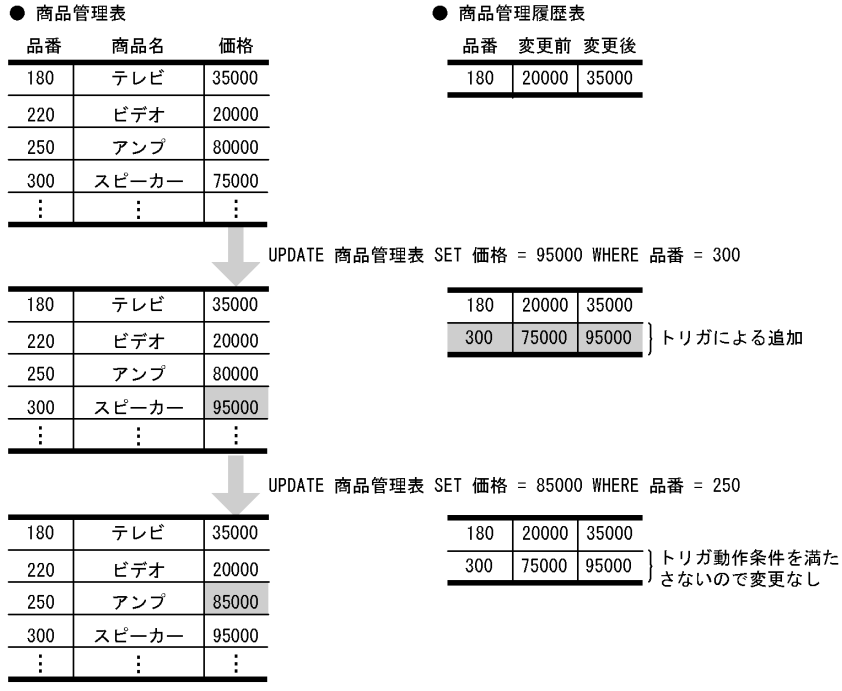
### (3) トリガの定義例

#### (a) トリガを使用した例

商品管理表の価格が更新された場合，変更前に比べ変更後の価格上昇が 10,000 円を超えるとき，商品管理

履歴表に更新前後の価格を挿入するトリガの定義例とトリガ動作を次に示します。

```
CREATE TRIGGER TR1          …トリガ名
       AFTER          …トリガ動作の実行時期
       UPDATE OF 価格          …トリガ動作の実行契機
       ON 商品管理表          …トリガを定義する表
       REFERENCING OLD ROW AS X1          …更新前の行に指定する名称
                   NEW ROW AS Y1          …更新後の行に指定する名称
       FOR EACH ROW          …行単位か文単位かの種別
       WHEN(Y1.価格 - X1.価格 > 10000)          …トリガ動作の探索条件
       INSERT INTO 商品管理履歴表 VALUES          …トリガSQL文
                      (X1.品番, X1.価格, Y1.価格)
```




### (b) トリガ動作に SQL 制御文（代入文）を使用した例

代入文は，指定した値を指定した列に代入する SQL です。トリガでは表に対する操作の前に動作するトリガ動作で代入文を使用できます。トリガ動作に代入文を使用すると列と列の関係付けができます。

社員表の職種列が更新されると，更新された内容によって手当て列の値を変更するトリガの定義例とトリガ動作を次に示します。

- 社員表に挿入される行に対して，職種が A の場合は給料の 8%，B の場合は 10%，A と B 以外の場合は 0%を手当ての列に設定するトリガ

```
CREATE TRIGGER 手当て設定トリガ1
       BEFORE
       INSERT
       ON 社員表
       REFERENCING NEW ROW AS X1
       FOR EACH ROW
       SET X1.手当て=CASE X1.職種 WHEN 'A' THEN X1.給料*0.08
                                  WHEN 'B' THEN X1.給料*0.1
                                  ELSE 0 END
```

- 職種と給料が更新された行に対して，職種がAの場合は給料の8%，Bの場合は10%，AとB以外の場合は0%を手当ての列に設定するトリガ

```
CREATE TRIGGER 手当て設定トリガ2
       BEFORE
       UPDATE OF 職種,給料
       ON 社員表
       REFERENCING NEW ROW AS X1
       FOR EACH ROW
       SET X1.手当て=CASE X1.職種 WHEN 'A' THEN X1.給料*0.08
                                  WHEN 'B' THEN X1.給料*0.1
                                  ELSE 0 END
```

● 社員表

| 社員番号 | 氏名 | 職種 | 給料 | 手当て |
|---|---|---|---|---|
| S99245 | 佐藤　一郎 | A | 190000 | 15200 |
| L98003 | 鈴木　浩一 | B | 220000 | 22000 |

INSERT INTO 社員表 VALUES ('R97023','田中　次郎','A','180000.0)

INSERT実行後

| 社員番号 | 氏名 | 職種 | 給料 | 手当て |
|---|---|---|---|---|
| S99245 | 佐藤　一郎 | A | 190000 | 15200 |
| L98003 | 鈴木　浩一 | B | 220000 | 22000 |
| R97023 | 田中　次郎 | A | 180000 | 14400 |

UPDATE 社員表 SET 職種='C' WHERE 社員番号='S99245'

UPDATE実行後

| 社員番号 | 氏名 | 職種 | 給料 | 手当て |
|---|---|---|---|---|
| S99245 | 佐藤　一郎 | C | 190000 | 0 |
| L98003 | 鈴木　浩一 | B | 220000 | 22000 |
| R97023 | 田中　次郎 | A | 180000 | 14400 |

〔説明〕

INSERT文が契機となり，手当て設定トリガ1が実行され，行が追加されました。INSERT文では手当てに0を設定していますが，代入文の結果が格納されます。

次に，UPDATE文が契機となり，手当て設定トリガ2が実行され，手当てが0に変更されました。

### (c) トリガ動作にSQL制御文（複合文）を使用した例

複合文は，複数のSQLをまとめて一つのSQLとして実行するSQLです。ある表に対する更新を契機に複数の表を更新する場合，トリガ動作に複合文を使用すると一つのトリガを定義するだけで複数の表を更新できます。

在庫マスタ表が更新された後，その更新内容を広島在庫及び仙台在庫に反映するトリガの定義例を次に示します。複合文を使用しない場合，トリガを二つ定義しなければなりません。

```
CREATE TRIGGER 地方在庫表更新トリガ
       AFTER
       UPDATE OF 在庫数
       ON 在庫マスタ
       REFERENCING NEW ROW 更新後
                   OLD ROW 更新前
       BEGIN
         UPDATE 広島在庫 SET 在庫数=更新後.在庫数
                WHERE 商品コード=更新前.商品コード;
         UPDATE 仙台在庫 SET 在庫数=更新後.在庫数
                WHERE 商品コード=更新前.商品コード;
       END
```

#### (d) トリガ動作に SQL 診断文（SIGNAL 文）を使用した例

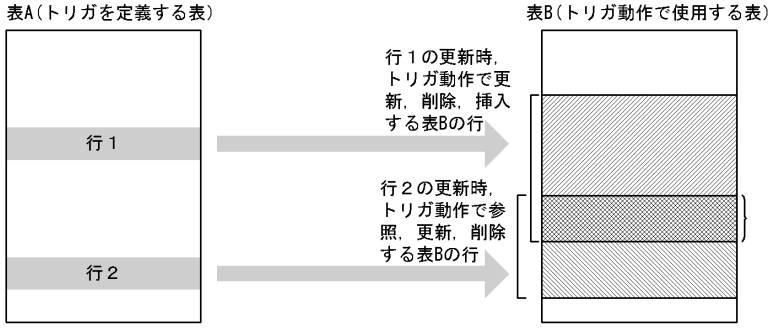
SIGNAL 文はエラーを発生させる SQL です。表への操作の前に SIGNAL 文を指定したトリガ動作を実行させれば，操作が不正な場合に SIGNAL 文を実行してその操作を抑止できます。

社員情報表を更新する前に，自分の情報以外を更新しようとした場合に SIGNAL 文でエラーを発生させ，更新を抑止するトリガの定義例を次に示します。

```
CREATE TRIGGER 更新抑止トリガ
       BEFORE
       UPDATE
       ON 社員情報
       REFERENCING OLD ROW AS X1
       WHEN(X1.社員名<>USER) SIGNAL SQLSTATE '99001'
```

## 12.5.3 トリガの使用上の注意

行単位トリガの定義によっては，トリガ契機となる SQL 実行時に HiRDB の内部処理に依存して異なる結果（更新の場合は異なる更新内容）になる場合があります。




〔説明〕

表 A の行 1 の更新時，トリガ動作で更新，削除，又は挿入する表 B の行と，行 2 の更新時，トリガ動作で参照，更新，又は削除する表 B の行に同一の行（重なり合った部分）があります。行 1 と行 2 の更新順序は HiRDB の内部処理に依存するため，この部分で異なる結果が得られる場合があります。

## 12.5.4 トリガの管理

### (1) トリガの定義

トリガを定義すると，トリガを定義する表を使用する関数，手続き及びトリガの SQL オブジェクトは無効になるため，再作成する必要があります。トリガを定義する前にディクショナリ表の SQL_ROUTINE_RESOURCES 表を参照すると，SQL オブジェクトが無効になる関数，手続き及びトリガを確認できます。無効になる SQL オブジェクトを確認して，再作成してください。

#### (a) トリガの定義によって SQL オブジェクトが無効になる関数，手続き及びトリガの確認

トリガの定義によって SQL オブジェクトが無効になる関数，手続き及びトリガを確認する SQL の例を次に示します。なお，無効になるのがトリガの場合は TRIGGER_NAME としてそのトリガ識別子が得られます。関数及び手続きの場合は TRIGGER_NAME が NULL になります。

```
SELECT DISTINCT B.ROUTINE_SCHEMA, B.ROUTINE_NAME, B.SPECIFIC_NAME,
```

```
A.TRIGGER_NAME
FROM MASTER.SQL_ROUTINE_RESOURCES B LEFT JOIN MASTER.SQL_TRIGGERS A
ON B.ROUTINE_SCHEMA=A.TRIGGER_SCHEMA
AND B.SPECIFIC_NAME=A.SPECIFIC_NAME
WHERE B.BASE_TYPE='R'
AND B.BASE_OWNER='トリガを定義する表の所有者の認可識別子'
AND B.BASE_NAME='トリガを定義する表の識別子'
AND (B.列名※ ='Y'
  OR (  B.INSERT_OPERATION IS NULL
    AND B.UPDATE_OPERATION IS NULL
    AND B.DELETE_OPERATION IS NULL))
```

注※

INSERT を契機とするトリガを定義すると無効になる SQL オブジェクトを検索する場合には，列名に INSERT_OPERATION を，UPDATE を契機とするトリガの場合は UPDATE_OPERATION を，DELETE を契機とするトリガの場合は DELETE_OPERATION を指定します。

## (2) トリガの SQL オブジェクトの再作成

トリガが使用している表，インデクスなどのリソースが定義，定義変更，又は削除されるとトリガの SQL オブジェクトは無効になります。また，トリガが使用している表にインデクスを定義したり，又はトリガが使用している表のインデクスを削除したりすると，トリガの SQL オブジェクトのインデクス情報が無効になります。

トリガの SQL オブジェクトが無効又は SQL オブジェクトのインデクス情報が無効の場合，トリガ契機となる SQL は実行できません。トリガの SQL オブジェクトの無効又は SQL オブジェクトのインデクス情報の無効を解除するには，定義系 SQL の ALTER TRIGGER 又は ALTER ROUTINE でトリガの SQL オブジェクトを再作成する必要があります。

### (a) トリガが使用しているリソースを確認する方法

ディクショナリ表の SQL_ROUTINE_RESOURCES，SQL_TRIGGER_USAGE，及び SQL_ROUTINE_PARAMS を参照すると，トリガが使用しているリソースの情報を得ることができます。

- トリガ動作条件で使用するリソースを確認する SQL の例

```
SELECT B.* FROM MASTER.SQL_TRIGGERS A, MASTER.SQL_TRIGGER_USAGE B
WHERE A.TRIGGER_SCHEMA='スキーマ名'
  AND A.TRIGGER_NAME='トリガ識別子'
  AND A.TRIGGER_SCHEMA=B.TRIGGER_SCHEMA
  AND A.TRIGGER_NAME=B.TRIGGER_NAME
```

- 新旧値別名を指定したトリガ SQL 中で使用する列リソースを確認する SQL の例

```
SELECT B.* FROM MASTER.SQL_TRIGGERS A, MASTER.SQL_ROUTINE_PARAMS B
WHERE A.TRIGGER_SCHEMA='スキーマ名'
  AND A.TRIGGER_NAME='トリガ識別子'
  AND A.TRIGGER_SCHEMA=B.ROUTINE_SCHEMA
  AND A.SPECIFIC_NAME=B.SPECIFIC_NAME
```

- 上記以外でトリガが使用するリソースを確認する SQL の例

```
SELECT B. * FROM MASTER.SQL_TRIGGERS A, MASTER.SQL_ROUTINE_RESOURCES B
WHERE A.TRIGGER_SCHEMA='スキーマ名'
  AND A.TRIGGER_NAME='トリガ識別子'
  AND A.TRIGGER_SCHEMA=B.ROUTINE_SCHEMA
  AND A.SPECIFIC_NAME=B.SPECIFIC_NAME
```

### (b) 表の列削除の前に，削除されるトリガを確認する方法

トリガ契機で指定されている列がすべて削除された場合，そのトリガは削除されます。表の列が削除される前に，削除されるトリガを確認する SQL の例を次に示します。

```
SELECT A.TRIGGER_SCHEMA, A.TRIGGER_NAME
  FROM MASTER.SQL_TRIGGERS A
WHERE A.N_UPDATE_COLUMNS>0
  AND A.TABLE_SCHEMA='列削除する表の所有者の認可識別子'
  AND A.TABLE_NAME='列削除する表の表識別子'
  AND NOT EXISTS(SELECT * FROM MASTER.SQL_TRIGGER_COLUMNS B
                 WHERE B.TRIGGER_SCHEMA=A.TRIGGER_SCHEMA
                   AND B.TRIGGER_NAME=A.TRIGGER_NAME
                   AND B.TABLE_SCHEMA=A.TABLE_SCHEMA
                   AND B.TABLE_NAME=A.TABLE_NAME
                   AND B.COLUMN_NAME NOT IN('削除する列名',…))
```

### (c) 表，インデクスなどの定義，定義変更，削除の前に SQL オブジェクトが無効又は SQL オブジェクトのインデクス情報が無効になる関数，手続き及びトリガを確認する方法

表，インデクスなどを定義，定義変更，又は削除する前に，SQL オブジェクトが無効又は SQL オブジェクトのインデクス情報が無効になる関数，手続き及びトリガを確認する SQL の例を次に示します。なお，無効になるのがトリガの場合は TRIGGER_NAME としてそのトリガ識別子が得られます。関数及び手続きの場合は TRIGGER_NAME が NULL になります。

● 表（ビュー表も含む）の定義変更及び削除，又はインデクス定義（インデクスを定義する表のスキーマ名と表識別子を指定）の場合

```
SELECT DISTINCT B.ROUTINE_SCHEMA, B.ROUTINE_NAME, B.SPECIFIC_NAME,
A.TRIGGER_NAME
  FROM MASTER.SQL_ROUTINE_RESOURCES B LEFT JOIN MASTER.SQL_TRIGGERS A
  ON B.ROUTINE_SCHEMA=A.TRIGGER_SCHEMA
    AND B.SPECIFIC_NAME=A.SPECIFIC_NAME
WHERE B.BASE_TYPE IN('R','V')
  AND B.BASE_OWNER='表（ビュー）の所有者の認可識別子'
  AND B.BASE_NAME='表（ビュー）の表識別子'
```

● インデクス削除の場合

```
SELECT DISTINCT B.ROUTINE_SCHEMA, B.ROUTINE_NAME, B.SPECIFIC_NAME,
A.TRIGGER_NAME
  FROM MASTER.SQL_ROUTINE_RESOURCES B LEFT JOIN MASTER.SQL_TRIGGERS A
  ON B.ROUTINE_SCHEMA=A.TRIGGER_SCHEMA
    AND B.SPECIFIC_NAME=A.SPECIFIC_NAME
WHERE B.BASE_TYPE ='I'
  AND B.BASE_OWNER='インデクスの所有者の認可識別子'
  AND B.BASE_NAME='インデクスの識別子'
```

● 関数，又は手続き削除の場合

```
SELECT DISTINCT B.ROUTINE_SCHEMA, B.ROUTINE_NAME, B.SPECIFIC_NAME,
A.TRIGGER_NAME
  FROM MASTER.SQL_ROUTINE_RESOURCES B LEFT JOIN MASTER.SQL_TRIGGERS A
  ON B.ROUTINE_SCHEMA=A.TRIGGER_SCHEMA
    AND B.SPECIFIC_NAME=A.SPECIFIC_NAME
WHERE B.BASE_TYPE ='P'
  AND B.BASE_OWNER='関数又は手続きの所有者の認可識別子'
  AND B.BASE_NAME='ルーチン識別子'
```

● トリガ削除の場合

```
SELECT DISTINCT B.ROUTINE_SCHEMA, B.ROUTINE_NAME, B.SPECIFIC_NAME,
A.TRIGGER_NAME
  FROM MASTER.SQL_ROUTINE_RESOURCES B LEFT JOIN MASTER.SQL_TRIGGERS A
  ON B.ROUTINE_SCHEMA=A.TRIGGER_SCHEMA
    AND B.SPECIFIC_NAME=A.SPECIFIC_NAME
WHERE B.BASE_TYPE ='T'
  AND B.BASE_OWNER='トリガの所有者の認可識別子'
  AND B.BASE_NAME='トリガ識別子'
```

● スキーマ削除の場合

```
SELECT DISTINCT B.ROUTINE_SCHEMA, B.ROUTINE_NAME, B.SPECIFIC_NAME,
```

```
  A.TRIGGER_NAME
    FROM MASTER.SQL_ROUTINE_RESOURCES B LEFT JOIN MASTER.SQL_TRIGGERS A
    ON B.ROUTINE_SCHEMA=A.TRIGGER_SCHEMA
      AND B.SPECIFIC_NAME=A.SPECIFIC_NAME
  WHERE B.BASE_OWNER='スキーマ名'
```

● ユーザ定義型削除の場合

```
  SELECT B.ROUTINE_SCHEMA, B.ROUTINE_NAME, B.SPECIFIC_NAME, A.TRIGGER_NAME
    FROM MASTER.SQL_ROUTINE_RESOURCES B LEFT JOIN MASTER.SQL_TRIGGERS A
    ON B.ROUTINE_SCHEMA=A.TRIGGER_SCHEMA
      AND B.SPECIFIC_NAME=A.SPECIFIC_NAME
  WHERE B.BASE_NAME='削除するデータ型識別子'
    AND B.BASE_TYPE='D'
  UNION
  SELECT B.ROUTINE_SCHEMA, B.ROUTINE_NAME, B.SPECIFIC_NAME, A.TRIGGER_NAME
    FROM MASTER.SQL_ROUTINES C INNER JOIN MASTER.SQL_ROUTINE_RESOURCES B
    ON C.SPECIFIC_NAME=B.BASE_NAME
   LEFT JOIN MASTER.SQL_TRIGGERS A
    ON B.ROUTINE_SCHEMA=A.TRIGGER_SCHEMA
      AND B.SPECIFIC_NAME=A.SPECIFIC_NAME
  WHERE C.ROUTINE_ADT_OWNER='削除するユーザ定義型の所有者の認可識別子'
  AND C.ROUTINE_ADT_NAME='削除するデータ型識別子'
  AND B.BASE_TYPE='P'
```

**(d) 表，インデクスなどの定義，定義変更，削除の後に SQL オブジェクトが無効又は SQL オブジェクトのインデクス情報が無効になった関数，手続き及びトリガを確認する方法**

表，インデクスなどを定義，定義変更，又は削除した後に，SQL オブジェクトが無効又は SQL オブジェクトのインデクス情報が無効になったトリガを確認するには，それぞれディクショナリ表の SQL_TRIGGER 表の TRIGGER_VALID 列，INDEX_VALID 列を参照します。TRIGGER_VALID 列の内容が 'N' の場合，そのトリガの SQL オブジェクトは無効になっています。また，INDEX_VALID 列の内容が 'N' の場合，そのトリガの SQL オブジェクトのインデクス情報は無効になっています。

表，インデクスなどを定義，定義変更，又は削除した後に，SQL オブジェクトが無効又は SQL オブジェクトのインデクス情報が無効になった関数，手続き及びトリガを確認する SQL の例を次に示します。なお，無効になるのがトリガの場合は TRIGGER_NAME としてそのトリガ識別子が得られます。関数及び手続きの場合は TRIGGER_NAME が NULL になります。

```
SELECT 'TRIGGER', TRIGGER_SCHEMA AS "SCHEMA", TRIGGER_NAME AS "NAME",
 TRIGGER_VALID AS "OBJECT_VALID", INDEX_VALID
  FROM MASTER.SQL_TRIGGERS
WHERE TRIGGER_VALID='N' OR INDEX_VALID='N'
UNION
SELECT 'ROUTINE', ROUTINE_SCHEMA, ROUTINE_NAME, ROUTINE_VALID, INDEX_VALID
  FROM MASTER.SQL_ROUTINES
WHERE ROUTINE_VALID='N' OR INDEX_VALID='N'
```

## 12.5.5 障害時の回復方法

トリガのソースはデータディクショナリ用 RD エリアに，トリガの SQL オブジェクトはデータディクショナリ LOB 用 RD エリアに格納されます。データディクショナリ用 RD エリアのログ取得モードは ALL，データディクショナリ LOB 用 RD エリアのログ取得モードは PARTIAL です。そのため，障害が発生した場合，ソースはバックアップとログを入力して回復すれば最新時点に回復できますが，SQL オブジェクトはバックアップ時点にしか回復できません。そこで次の運用をする必要があります。

● 常に最新のバックアップを取得しておく
常にデータディクショナリ LOB 用 RD エリアのバックアップを取得し，障害発生時に最新のバックアップから回復します。pdcopy コマンドで -M x，又は -M r を指定してください。

バックアップの取得方法については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

- トリガの SQL オブジェクトを再作成する
  データディクショナリ LOB 用 RD エリアの最新のバックアップがない場合，そのデータディクショナリ LOB 用 RD エリアを pdmod コマンドで再初期化します。その後，ALTER ROUTINE に ALL を指定して実行し，全トリガの SQL オブジェクトを再作成します。

# 12.6　ビュー表の作成

実際にデータベースに格納している表（**実表**）から特定の行や列を選択して，新たに定義した仮想の表（**ビュー表**）を作成できます。HiRDB External Data Access 機能を使用する場合は，外部表からビュー表を作成できます。

## (1) 実表とビュー表の関係

実表とビュー表の関係を次の図に示します。

**図 12-18　実表とビュー表の関係**

● 実表（ZAIKO）

| 商品コード | 商品名 | 色 | 単価 | 在庫量 |
|---|---|---|---|---|
| SCODE | SNAME | COL | TANKA | ZSURYO |
| 101L | ブラウス | 青 | 3500 | 62 |
| 101M | ブラウス | 白 | 3500 | 85 |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| 412M | セーター | 赤 | 8400 | 22 |
| 591L | ソックス | 赤 | 250 | 300 |
| 591M | ソックス | 青 | 250 | 90 |
| 591S | ソックス | 白 | 250 | 280 |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |



● ビュー表（VZAIKO）

| 商品コード | 在庫量 | 単価 |
|---|---|---|
| SCODE | ZSURYO | TANKA |
| 591L | 300 | 250 |
| 591M | 90 | 250 |
| 591S | 280 | 250 |

〔説明〕

実表（ZAIKO）から，商品名（SNAME）がソックスの行で，商品コード（SCODE），在庫量（ZSURYO）及び単価（TANKA）の列で構成されるビュー表（VZAIKO）を作成した例です。

商品を扱うある支店では，「商品名」が「ソックス」で「商品コード」「在庫量」「単価」のデータを参照だけすればよい場合，実表（ZAIKO）にはアクセス不可とし，ビュー表（VZAIKO）に対して参照だけを許す権限（SELECT 権限）を設定します。これによって，必要な情報だけを参照でき，データは保護されます。

● HiRDB External Data Access 機能を使用する場合

外部表からも実表と同様にビュー表を作成できます。また，外部表から作成したビュー表の更新もできます。

## (2) ビュー表を作成したときの効果

ビュー表を作成したときの効果を次に示します。

**機密性の向上**

自分の表の機密保護を図りたい場合にビュー表を作成します。ビュー表を作成することで，表の特定の列と行だけを表示できます。このため，アクセス権限をビュー表に設定することで，行又は列レベルの機密保護が図れます。

**操作性の向上**

- 複雑な問い合わせ指定をして表を検索しなくてもよいように，複雑な問い合わせ指定をした場合に検索できるデータであらかじめビュー表を作成します。これによって，表を参照する操作が手軽になります。
- ビュー表を通して実表を参照したり更新したりできます。これによって，実表の定義が変更されて

も SQL 文を変更する必要はなく，場合によってはビュー表の定義を変更する必要がなくなります。

## (3) ビュー表の作成方法

定義系 SQL の CREATE VIEW を実行してビュー表を作成します。CREATE VIEW で次に示すビュー表を定義できます。

- 実表，外部表の特定の行及び列を選択したビュー表
- 実表，外部表の列から集合関数，日付演算，時刻演算，連結演算，スカラ関数，又は四則演算で求めた値を列とするビュー表
- 最大 64 個の実表を基にした一つのビュー表
- グループ分け検索の結果を基にしたビュー表
- ほかのユーザが所有する実表又は外部表を基にしたビュー表（ほかのユーザが所有する実表又は外部表の SELECT 権限を与えられている場合に限る）

**規則**

1. 一つのビュー表には 30,000 個まで列を定義できます。
2. ビュー表に対して列の追加，及びインデクスを定義できません。
3. 自分の所有する実表から定義したビュー表の所有者はビュー表に対するすべてのアクセス権限（行の検索，追加，削除，更新）を持ちます。
4. ほかのユーザが所有する実表又は外部表から定義したビュー表の所有者は，ビュー表に対するアクセス権限として，実表又は外部表に対して付与されているアクセス権限だけを持ちます。ただし，ビュー表の定義に次に示すどれかの指定をした場合は，機密保護機能を使用するかどうかに関係なく行の検索しかできません。

- ビュー表の列に実表又は外部表の同じ列を複数指定した場合
- ビュー表の列に定数，USER，CURRENT_DATE，CURRENT_TIME，四則演算，日付演算，時刻演算，連結演算，又はスカラ関数の結果を指定した場合
- 複数の実表又は外部表を指定した場合
- DISTINCT，集合関数（COUNT(＊)，AVG，MAX，MIN，SUM），グループ分け（GROUP BY 句），又はグループ条件（HAVING 句）を指定した場合

  機密保護機能を使用しない場合，これら以外のビュー表はほかのユーザが自由に更新できます。ただし，読み込み専用のビュー表（READ ONLY 指定）の場合，機密保護機能を使用しなくてもほかのユーザがビュー表を更新できません。

## (4) ビュー表の削除方法

定義系 SQL の DROP VIEW を実行してビュー表を削除します。ビュー表を削除すると，削除したビュー表に関するすべてのアクセス権限が取り消されます。

# 12.7 FIX 属性の指定

FIX **属性**とは，行長が固定の表に付ける属性のことです。

## (1) FIX 属性を指定したときの効果

表に FIX 属性を指定したときの効果を次に示します。

**性能の向上**

- 任意の列を取り出す性能が，列の定義順序に関係なく一定になります。また，FIX 属性を指定しない場合と比べて，列の取り出し時間を短縮できます。
- UAP で行単位インタフェースが使用できるため，列数が多くてもアクセス性能を向上できます。

**操作性の向上**

FIX 属性の表の列を更新する場合に入力データにナル値があったときは，エラーとしてチェックアウトできます。

**ディスク所要量の削減**

FIX 属性を指定していない表と比べて，物理的な行長が 1 列で 2 バイト短くなります。このため，列数の多い表の場合はディスク所要量を削減できます。

## (2) 適用基準

次に示す条件をすべて満たす場合は，表定義時に FIX 属性を指定します。

- 列を追加しない
- ナル値を持つ列がない
- 可変長の列がない

また，上記の条件を満たさない場合でも，次に示すことを検討してください。

- 0（数値データ）又は空白（文字データ）を使用して，ナル値を不要にしてください。ただし，ナル値は探索条件や集合関数での扱いが，ほかの値と異なるため注意してください。
- 最大値の小さい可変長データや実長の取り得る範囲が狭い可変長データを固定長にしてください。ただし，探索条件での扱いが異なるため注意してください。

## (3) 指定方法

表に FIX 属性を指定するには，定義系 SQL の CREATE TABLE で FIX を指定（CREATE FIX TABLE と指定）します。

# 12.8 主キー（プライマリキー）の指定

表中の行を一意（ユニーク）に識別するためのキーとして**主キー**があります。主キーを定義すると，指定した列に対してインデクスが作成されます。

## (1) 主キーを定義したときの効果

主キーを定義した列には，**一意性制約**と**非ナル値制約**が適用されます。一意性制約とは，キー（列又は複数の列の組）中のデータの重複を許さない（キー中のデータが常に一意である）制約のことです。非ナル値制約とは，キー中の各列の値にナル値を許さない制約のことです。

## (2) 適用基準

行を一意に識別できる列に主キーを定義します。表中に行を一意に識別できる列又は列の組（**候補キー**）が複数ある場合，その候補キーの中から主キーを選んでください。表中のキーの中で意味的に最も重要で，かつ一意性制約及び非ナル値制約を設定したいキーに主キーを定義します。

## (3) 指定方法

表に主キーを定義するには，定義系 SQL の CREATE TABLE の PRIMARY KEY オプションを指定します。

# 12.9 クラスタキーの指定

**クラスタキー**とは，特定の列の値の昇順又は降順に行を格納するためのキーとして指定した列のことです。表の一つの列又は複数の列にクラスタキーを指定しておくと，クラスタキーの昇順又は降順に行を格納できます。

クラスタキーを定義すると，指定した列に対してインデクスが作成されます。

## (1) クラスタキーを定義したときの効果

表にクラスタキーを指定したときの効果を次に示します。

**性能の向上**

範囲を指定して行の検索，更新，削除などをするとき又はクラスタキー順に検索，更新などをするときに，入出力時間を削減できます。

**操作性の向上**

- UNIQUE 指定のクラスタキーを定義すると，そのクラスタキーには一意性制約が適用されます。このため，行を挿入するときに，クラスタキーを構成している列の値がすべての行で重複しないようにできます。なお，フレキシブルハッシュ分割表には，UNIQUE 指定のクラスタキーを定義できません。
- PRIMARY 指定のクラスタキーを定義すると，そのクラスタキーには一意性制約及び非ナル値制約が適用されます。このため，行を挿入するときに，クラスタキーを構成している列の値がすべての行で重複しないようにできます。また，クラスタキーを構成している列の値にナル値を格納しないようにできます。なお，フレキシブルハッシュ分割表には，PRIMARY 指定のクラスタキーを定義できません。
- 表を作成するときに，入力データがクラスタキーの昇順又は降順に並んでいるかどうかをデータベース作成ユティリティ（pdload）で確認できます。
- 表を再編成するときに，アンロードした行のクラスタキーと，リロードするクラスタキーが一致しているかどうかをデータベース再編成ユティリティ（pdrorg）で確認できます。

## (2) 適用基準

クラスタキーを指定するとよい場合を次に示します。

- キーの昇順又は降順にデータを蓄積する業務で，キー順にアクセスする業務が多い場合
- キーを変更しない表の場合
- 行長が固定の表の場合

## (3) 指定方法

表にクラスタキーを定義するには，定義系 SQL の CREATE TABLE で CLUSTER KEY オプションを指定します。

## (4) 設計上の考慮点

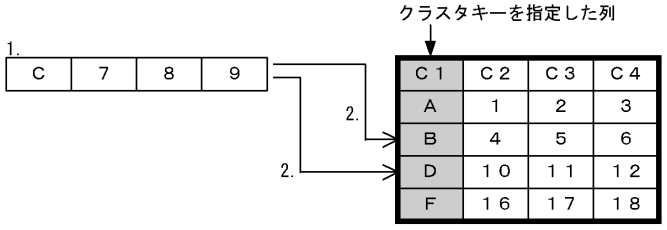
データを追加した後の検索効果を上げるため，表を格納しているページ内に未使用領域を設定しておきます。設定方法については，「14.3　ページ」を参照してください。

## (5) 注意

- クラスタキーを構成する列の値は更新できません。
- クラスタキーを構成する列にナル値は挿入できません。
- クラスタキーを指定した表にデータを追加するとき，追加しようとするキー値に近接するキー値を持っ

たページを探すためのオーバヘッドが発生します。この概要を次の図に示します。

**図 12-19 クラスタキーを指定した表にデータを追加するときのオーバヘッドの概要**




〔説明〕

1. 列 C を持つデータを追加します。
2. 列 C に近接するキー値を探すオーバヘッドが発生します。

# 12.10 サプレスオプションの指定

**サプレスオプション**とは，表中のデータの一部を省略して，実際のデータ長よりも短くして格納するオプションのことです。

サプレスオプションを指定した場合には，データ格納時に，表中の DECIMAL データの有効けた（先頭の 0 の部分を除いたけた）部分及び格納データ長だけを格納します。

## (1) サプレスオプションを指定したときの効果

サプレスオプションを指定したときの効果を次に示します。

**性能の向上**

- 実際のデータ長よりも短くしてデータを格納できるため，ディスク所要量を削減できます。
- ディスク所要量の削減で，全件検索などの検索処理での入出力時間が短縮できます。

## (2) 適用基準

次に示す場合にサプレスオプションを指定することをお勧めします。

- 表中に DECIMAL データが多く，有効けた数が多い場合
- 全件検索などの検索業務が多く，更新業務が少ない場合

## (3) 指定方法

サプレスオプションを指定するには，定義系 SQL の CREATE TABLE で SUPPRESS オプションを指定します。

## (4) 注意

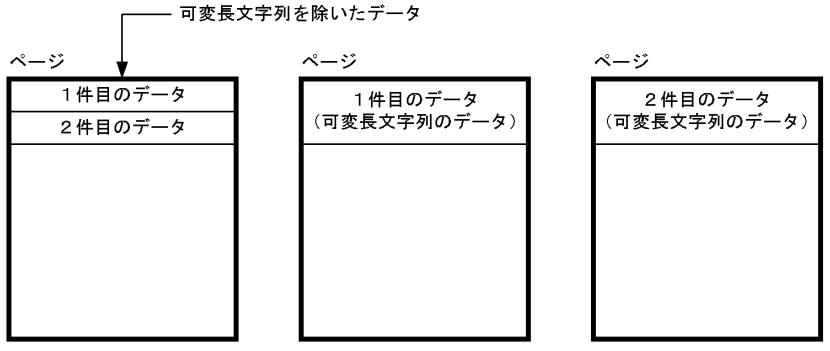
- DECIMAL データの有効けた数が定義長と等しい場合又は定義長が 1 の場合には，「定義長＋ 1」の長さで格納されます。このため，サプレスオプション指定時よりも格納するデータ長が長くなります。
- FIX 属性を指定している表では，サプレスオプションを指定できません。

# 12.11 ノースプリットオプションの指定

表に次に示すデータ型が定義されていて，次に示すデータ型の実際のデータ長が 256 バイト以上の場合，1 行のデータを複数のページに格納します。このときのデータ格納方式を次の図に示します。

- VARCHAR
- MVARCHAR
- NVARCHAR

**図 12-20 可変長文字列の実際のデータ長が 256 バイト以上の場合のデータ格納方式**



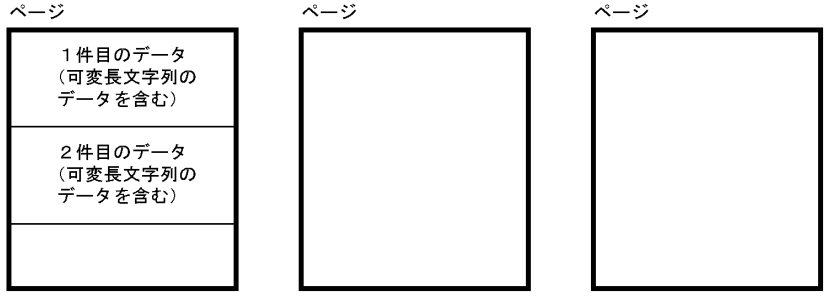

〔説明〕

可変長文字列を除いたデータと可変長文字列のデータは，異なるページに格納されます。このため，データの格納効率が低下します。このような場合に，**ノースプリットオプション**を指定してデータの格納効率を向上してください。

## (1) 適用基準

**ノースプリットオプション**を指定すると，可変長文字列の実際のデータ長が 256 バイト以上であっても，1 行を 1 ページに格納します。ノースプリットオプションを指定したときのデータ格納方式を次の図に示します。

**図 12-21 ノースプリットオプションを指定したときのデータ格納方式**




〔説明〕

1 行の全データを同じページに格納します。このため，データの格納効率がノースプリットオプションを指定しないときに比べて向上します。

### (2) 指定方法

ノースプリットオプションを指定するには，定義系 SQL の ALTER TABLE，CREATE TABLE 又は CREATE TYPE で NO SPLIT オプションを指定します。

### (3) 注意

- 1 行のデータ長の合計がページ長を超える場合は，ノースプリットオプションを指定してもスプリット（1 行のデータを複数のページに格納）します。
- 可変長文字列の実際のデータ長が 255 バイト以下の場合にノースプリットオプションを指定すると，指定しないときに比べて列データ長が 1 バイト長くなります。
- ノースプリットオプションを指定すると，可変長文字列の実際のデータ長が 256 バイト以上であっても分岐しないため，ノースプリットオプションを指定しない場合と比べて 1 ページに格納される行数が少なくなります。このため，インデクススキャンでノースプリットオプションが適用された可変長文字列型の列データを取り出さない検索をすると，ノースプリットオプションを指定しないときに比べてアクセスするページ数が多くなり，検索性能が低下することがあります。ただし，キースキャン及びテーブルスキャンの場合は影響はありません。

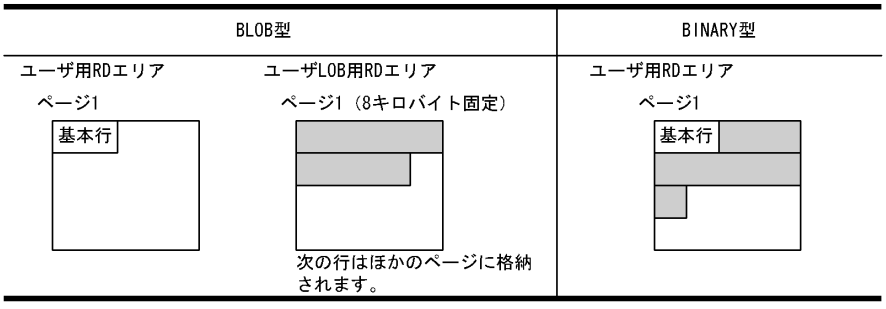
# 12.12 バイナリデータ列の指定

文書，画像，音声などの可変長バイナリデータを格納する列に指定するデータ型には，次の二つがあります。

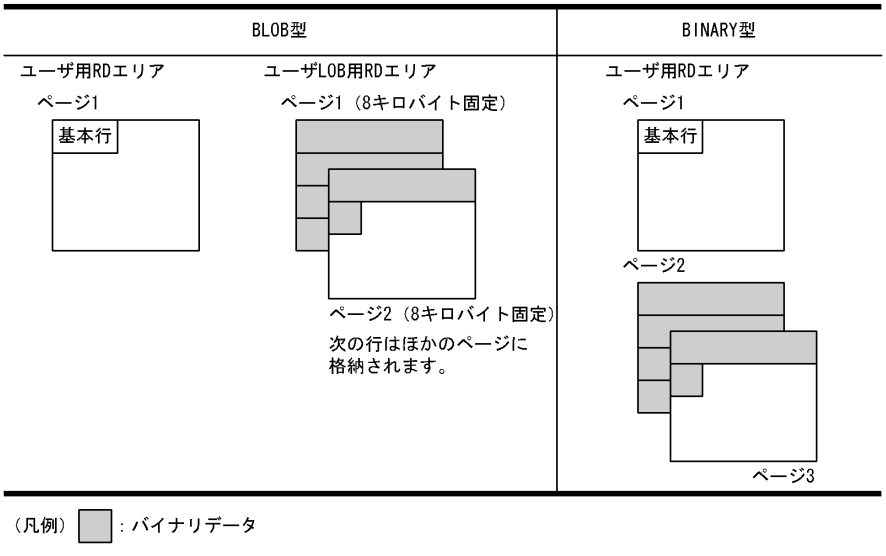
- BLOB 型（BLOB 型を指定した列を LOB 列といいます）
- BINARY 型

それぞれのデータの格納方法の比較を次に示します。

**データ長（基本行＋バイナリデータ長）が 1 ページ以下の場合**




**データ長（基本行＋バイナリデータ長）が 1 ページより大きい場合**




〔説明〕

BLOB 型を指定した列はほかの属性の列とは別に，ページ長が 8 キロバイト固定のユーザ LOB 用 RD エリアに格納します。

BINARY 型を指定した列は表を構成するすべての列データが 1 ページに格納できる場合，実際のデータ長に関係なく 1 行を 1 ページ内に格納します。ただし，1 行が 1 ページに格納できない場合は，1

行が複数ページになり，バイナリ専用セグメントに格納します。

## 12.12.1 BLOB 型

### (1) 設計上の考慮点

- BLOB 型を指定する場合はユーザ LOB 用 RD エリアを作成してください。
- BLOB 型のデータは 8 キロバイトバウンダリされるので，データの格納効率は BINARY 型が優位です。ただし，8 キロバイトのバウンダリが無視できるような長大データサイズになると余り差はありません。

### (2) 指定方法

定義系 SQL の CREATE TABLE の列のデータ型で，BLOB 型を指定します。

### (3) 注意

BLOB 型は次の項目には使用できません。

- FIX 属性の表
- インデクスの定義
- 分割キー

## 12.12.2 BINARY 型

### (1) 設計上の考慮点

BINARY 型は，BLOB 型に比べて 1 ページに格納する行数が減るため，ある列を探索条件に指定して，BINARY 型を指定したデータを取り出さない検索を実行した場合，BLOB 型に比べて入出力回数が増えます。そのため，検索性能が低下することがあります。ただし，BINARY 型以外の列にインデクスを定義してインデクススキャンにすると BINARY 型と BLOB 型の性能差はなくなります。

### (2) 指定方法

定義系 SQL の CREATE TABLE の列のデータ型で，BINARY 型を指定します。

### (3) 注意

- BINARY 型は次の項目には使用できません。
  - FIX 属性の表
  - インデクスの定義
  - 分割キー
  - 外への参照列
- バイナリデータ長が 1 ページより大きい場合，基本行を格納するセグメントと異なるバイナリ専用セグメントに格納します。データ長が 1 ページより大きいバイナリデータ格納時，基本行を格納するセグメントに未使用ページがある場合でも，バイナリ専用セグメントに未使用ページがなければ RD エリアが容量不足となる場合があります。RD エリアが容量不足になった場合，データベース解析ユティリティ（pddbst）でバイナリ専用セグメント，及び基本行を格納するセグメントのそれぞれの使用状況を参照することで，原因がバイナリデータによるものか，又は基本行のデータによるものかが確認できます。
- BINARY 型の列を定義する場合は，列の定義長（列の最大長）の見積もりを詳細に行ってください。列の定義長を不要に大きくすると，列の定義長に比例してメモリを使用するデータロードを実行できなくなることがあります（BINARY 列を定義している表のデータロードを実行する場合，BINARY 列の

定義長分のメモリを確保します)。

## 12.12.3 BLOB 型と BINARY 型の使い分け

バイナリデータの運用方法によってお勧めするデータ型を次の表に示します。

**表 12-4 バイナリデータの運用による推奨データ型**

| バイナリデータの運用方法 | | 推奨するデータ型の平均データサイズ | | 説明 |
|---|---|---|---|---|
| | | 32,000 バイト以下の場合 | 32,000 バイトより大きい場合 | |
| 射影列にバイナリデータ列を指定する頻度 | 高い | BINARY | BLOB | • 32,000 バイト以下の場合，BINARY 型の方がブロック転送もでき，行全体のデータが近くに集まって配置されるので，性能的に優位です。<br>• 32,000 バイトより大きい場合，BLOB 型の方が長大データに特化した内部メモリ削減などの処理をするので，性能的に優位です。 |
| | 低い | BLOB ※ | BLOB | BLOB 型の方が BINARY 型より基本行のデータサイズが小さいので，データ件数が多くなれば BLOB 型が性能的に優位です。ただし，BINARY 型以外の列にインデクスを定義してインデクススキャンにすることで BINARY 型と BLOB 型の性能差はなくなります。また，インデクススキャンにすることを推奨しますが，セグメントサイズを大きくすることでも性能差を小さくできます。 |
| SQL 記述の柔軟性 | | BINARY | 同等 | 32,000 バイト以下の場合，BINARY 型の SQL 記述はインデクス定義を除いて，ほぼ VARCHAR と同等の記述ができます。そのため，BLOB 型より SQL 記述範囲が広がります。詳細は，マニュアル「HiRDB Version 8 SQL リファレンス」を参照してください。 |
| データ格納効率 | | BINARY | | 格納効率は BINARY 型が優位です。ただし，8 キロバイトのバウンダリが無視できるような長大データサイズになると余り差はありません。 |
| 追加更新を頻繁にする場合 | | BLOB | | 連結演算するデータサイズが大きいほど BLOB 型が性能的に優位です。 |
| 部分抽出を頻繁にする場合 | | 同等 | BLOB | BINARY 型の格納済みデータサイズが大きいデータに対して部分抽出をすると非常に性能が悪くなります。さらに，部分抽出回数を多くすると，性能は劣化します。大きなデータに対して部分抽出を頻繁にする必要がある場合には，BLOB 型をお勧めします。 |
| 運用性を重視する場合 | | BINARY | | BLOB 型はユーザ LOB 用 RD エリアをバックアップするなどの特別な運用が必要になります。 |
| 上記使用方法で判断できない場合，又は将来的に方針変更の可能性があり，判断できない場合 | | BINARY ※ | BLOB | 32,000 バイトより大きいデータを扱う場合は BLOB 型をお勧めします。比較的データサイズが小さい場合は BINARY 型をお勧めします。 |

注※

データサイズがページサイズに近い大きさで分岐しない場合，BINARY 型で大量のテーブルスキャンをすると，BLOB 型に比べて非常に性能が悪くなります。これを防ぐため，テーブルスキャンからイ

ンデクススキャンへ変更してください。また，インデクススキャンにすることを推奨しますが，セグメントサイズを大きくすることでも性能差を小さくできます。

# 12.13 文字集合の指定

文字集合とは，文字データに対する属性です。文字集合は，次の三つの属性を持ちます。

- **使用形式**
  文字を表現する規約のことです。例えば，ある文字集合では A を 1 バイトのコード X'41' と表現しますが，別の文字集合では，これを X'C1' と表現します。このように，文字を表現する規約のことを使用形式といいます。
- **文字レパートリ**
  表現できる文字の集合のことです。例えば，ある文字集合ではバックスラッシュを表現できますが，別の文字集合ではこれを表現できません。このように，表現できる文字の集まりを文字レパートリといいます。
- **既定の照合順**
  二つの文字列データを比較する場合の規約のことです。例えば，ある文字集合では '1' ＞ 'A' ですが，別の文字集合では 'A' ＞ '1' という照合順になります。どの文字集合にも，既定の照合順があります。

なお，文字集合指定を省略して仮定された文字集合を，既定文字集合といいます。

## (1) 文字集合を定義したときの効果

文字集合を定義すると，表の列ごとに異なる文字集合の文字列データを格納できます。これによって，文字集合に EBCDIK を指定した場合は，VOS3 システムから HiRDB に移行したときに，データベースに格納していた文字データを VOS3 システムの文字列データの照合順で検索，代入，比較ができます。また，文字集合に UTF16 を指定した場合は，UTF-16 で文字データの検索，代入，比較ができます。

## (2) HiRDB で使用できる文字集合

HiRDB で使用できる文字集合を次に示します。

- EBCDIK
  EBCDIK を使用する場合には，HiRDB セットアップ時，文字コード種別に sjis を指定します。
- UTF16
  UTF16 を使用する場合には，HiRDB セットアップ時，文字コード種別に utf-8 を指定します。

## (3) 指定方法

文字集合は文字データ型に指定します。文字集合を指定する場合の形式や規則については，マニュアル「HiRDB Version 8 SQL リファレンス」を参照してください。

## (4) 注意事項

- 文字集合を指定した列にデータロードをする場合，入力データファイルはバイナリ形式にしてください。バイナリ形式以外のファイル形式の場合，データロードできません。
- 文字集合を指定した列がある表を再編成する場合，アンロードデータファイルはバイナリ形式にしてください。バイナリ形式以外のファイル形式の場合，アンロード，及びリロードはできません。
- 入力データの文字集合と列の文字集合が異なる場合はデータロードできません。
- 文字集合に UTF16 を指定した場合，データベースに格納されるデータはビッグエンディアン形式になります。文字集合に UTF16 を指定した列に対して，埋込み変数，又は？パラメタを使用して操作する場合は，埋込み変数，？パラメタに指定する値をビッグエンディアン形式にしてください。これらをリトルエンディアン形式にした場合，文字コード変換をする必要があるため，性能が悪くなります。

# 12.14　WITHOUT ROLLBACK オプションの指定

WITHOUT ROLLBACK オプションとは，表に対しての更新処理（追加，削除を含む）の完了を契機に，その更新行に対する排他が解除され，それ以降はロールバックされなくなるオプションのことです。

## (1) WITHOUT ROLLBACK オプションを指定したときの効果

表に WITHOUT ROLLBACK オプションを指定したときの効果を次に示します。

**性能の向上**

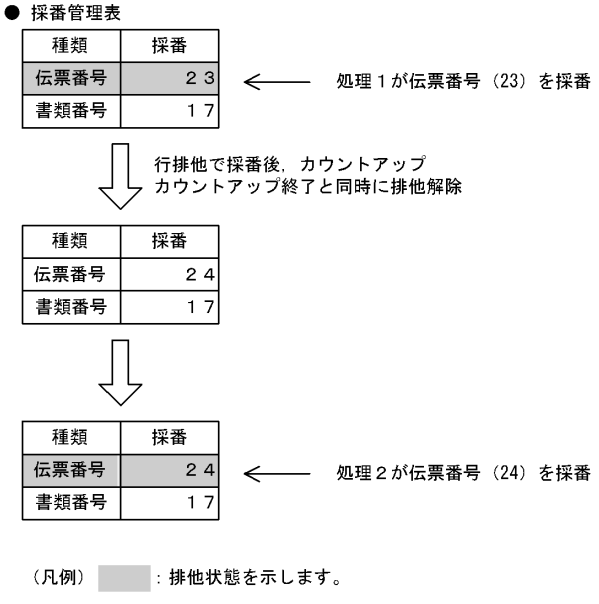
更新処理完了時に排他が解除されるので，排他待ちの発生を削減できます。

## (2) 適用基準

採番業務のような，更新処理が集中する表に適しています。

伝票番号や書類番号などの採番業務では，一つの表で番号を管理し，採番するごとにカウントアップするという方法が考えられます。しかし，この方法では処理が集中すると，COMMIT 発行まで排他が解除されないため，排他待ちが多く発生することが考えられます。このような処理をする場合，その表に WITHOUT ROLLBACK オプションを指定すると，カウントアップが終了した時点で排他が解除されるため，排他待ちの発生を削減できます。採番業務の例を次の図に示します。

**図 12-22　採番業務の例**




この例では，1 行で 1 種類の番号を管理しています。

図 12-22 に示した採番管理表の定義例を次に示します。

```
CREATE TABLE 採番管理表 (種類 NCHAR(4),
                         採番 INT)
                    :
               WITHOUT ROLLBACK
```

ただし，次に示す要因で欠番が発生する可能性がありますので，欠番が発生しても問題がない業務で使用してください。

- WITHOUT ROLLBACK オプションを指定して定義した表は，行を更新した時点でロールバックされなくなります。そのため，UAP や HiRDB が異常終了して再開始した場合，採番した番号を使用した業務の表は正しくロールバックされますが，WITHOUT ROLLBACK オプションの表はどの時点までロールバックされたのか分かりません。このような場合，採番はしたが，その番号が業務で使用されないことがあります。

### （3）注意

- データベース作成ユティリティ（pdload）及びデータベース再編成ユティリティ（pdrorg）をログ取得モードを指定して実行した場合，WITHOUT ROLLBACK オプションを指定した表でも，通常の表と同様にロールバックされます。
- 採番業務のように更新処理が集中する表の場合，専用の RD エリア及びグローバルバッファを割り当ててください。
- WITHOUT ROLLBACK オプションを指定している表には，行更新時にインデクス構成列に対して同値更新となる場合だけインデクスを作成できます。また，行挿入及び行削除時に行排他が解除されないので，通常の表と同様にロールバックされます。

# 12.15 改竄防止機能の指定

**改竄防止機能**とは，表の所有者も含め，すべてのユーザに対して表データの更新を禁止する機能です。この機能を適用することで，重要なデータを人為的ミスや不正な改竄から守れます。この機能が適用された表を**改竄防止表**といいます。改竄防止表に対する操作の実行可否を次の表に示します。

**表 12-5 改竄防止表に対する操作の実行可否**

| 操作 | 改竄防止表 | |
|---|---|---|
| | 行削除禁止期間指定あり | 行削除禁止期間指定なし |
| 挿入（INSERT） | ○ | ○ |
| 検索（SELECT） | ○ | ○ |
| 列単位の更新（UPDATE） | ○※1 | ○※1 |
| 行単位の更新（UPDATE） | × | × |
| 削除（DELETE） | ○※2 | × |
| 全行削除（PURGE TABLE） | × | × |
| 上記以外の操作系 SQL | ○ | ○ |

（凡例）
○：実行できます。
×：実行できません。

注※1
更新可能列だけ更新できます。

注※2
行削除禁止期間を経過しているデータだけ削除できます。行削除禁止期間を指定しないと，表データを削除できません。

**適用基準**
改竄防止機能は，表のデータを誤って，又は不当に更新されることを防止したい表に適用をお勧めします。

## 12.15.1 指定方法

定義系 SQL の CREATE TABLE で INSERT ONLY オプション（改竄防止オプション）を指定します。また，ALTER TABLE で INSERT ONLY オプションを指定して，既存の表を改竄防止表に定義変更することもできます。

表定義又は定義変更時に次の列を定義できます。

- **更新可能列**
  更新可能列を定義すると，列単位に，次の方法でデータを更新できます。
  - 常に更新できる（UPDATE を指定）
  - ナル値から非ナル値へ一度だけ更新できる（UPDATE ONLY FROM NULL を指定）

  更新可能列を定義できるのは，次の時点です。
  - CREATE TABLE 実行時
  - ALTER TABLE（CHANGE INSERT ONLY）実行前

- ALTER TABLE（ADD 列名），又は ALTER TABLE（CHANGE 列名）※実行時

注※

ALTER TABLE（CHANGE 列名）は，改竄防止表に対しては実行できません。既存の表を改竄防止表に定義変更する場合，事前に実行します。

- **挿入履歴保持列**

挿入履歴保持列を定義すると，**行削除禁止期間**を指定できます。行削除禁止期間を省略すると，表データは一切削除できません。また，表にデータがあると DROP TABLE が実行できないため（「12.15.2(1) 定義系 SQL」参照），行削除禁止期間を省略した場合，表及び表データは共に削除できません。そのため，データの保存期間が決まっている場合，又は決められる場合は行削除禁止期間を指定してください。

なお，データベース再編成ユティリティや pdrels コマンドでの RD エリアに対する運用に制限がある※ため，改竄防止表は 1 表 1RD エリアに格納することをお勧めします。

注※

データベース再編成ユティリティで改竄防止表の再編成をする場合，RD エリアのコマンド閉塞が必須です。また，データベース再編成ユティリティが異常終了した場合は，再編成が完了するまで RD エリアの閉塞が解除できないため，該当する RD エリアに別の表やインデクスが定義されている場合は，その表やインデクスが使用できなくなります。詳細は，「12.15.2　制限事項」を参照してください。

## 12.15.2　制限事項

改竄防止表はデータの更新及び削除ができない表のため，改竄防止表又は改竄防止表格納 RD エリアに対して，SQL 文，ユティリティ，及びコマンドの実行に制限があります。

### (1) 定義系 SQL

改竄防止表に対して実行できない定義系 SQL 文があります。制限がある定義系 SQL 文と制限事項を次の表に示します。

**表 12-6　制限がある定義系 SQL 文と制限事項**

| SQL 文 | 制限事項 |
|---|---|
| CREATE TABLE | すべての列に更新可能列属性を指定し，かつ改竄防止オプションを指定することはできません。 |
| ALTER TABLE | • 表名称，及び列名称は変更できません。<br>• データが存在する表に改竄防止機能は適用できません。既存の表に改竄防止機能を適用する手順については，「12.15.3　非改竄防止表から改竄防止表への変更」を参照してください。<br>• 改竄防止機能の解除はできません。<br>• 既存の列を更新可能列に変更したり，更新可能列を通常の列に変更したりすることはできません。<br>• 更新可能列は，改竄防止機能を適用する前に定義しておく必要があります※。<br>• 行削除禁止期間の設定，解除，及び期間の変更はできません。<br>• 行削除禁止期間が指定されている改竄防止表の場合，行削除禁止期間を指定した挿入履歴保持列は削除できません。<br>• 分割格納条件は変更できません。 |
| DROP TABLE | 改竄防止表にデータがある場合，実行できません。 |

注※

既存の表に更新可能列を指定して，改竄防止機能を適用するためには，列及び表に対してそれぞれ ALTER TABLE を実行する必要があります。次の手順で改竄防止機能を適用してください。

1. ALTER TABLE で，更新可能とする列の属性を更新可能列に変更
2. ALTER TABLE（CHANGE INSERT ONLY）で，改竄防止表にする表に対して改竄防止機能を適用

## (2) ユティリティ

改竄防止表，又は改竄防止表格納 RD エリアに対して，ユティリティ運用が制限されます。制限されるユティリティと制限事項を次の表に示します。記載されていないユティリティについては，特に制限はありません。

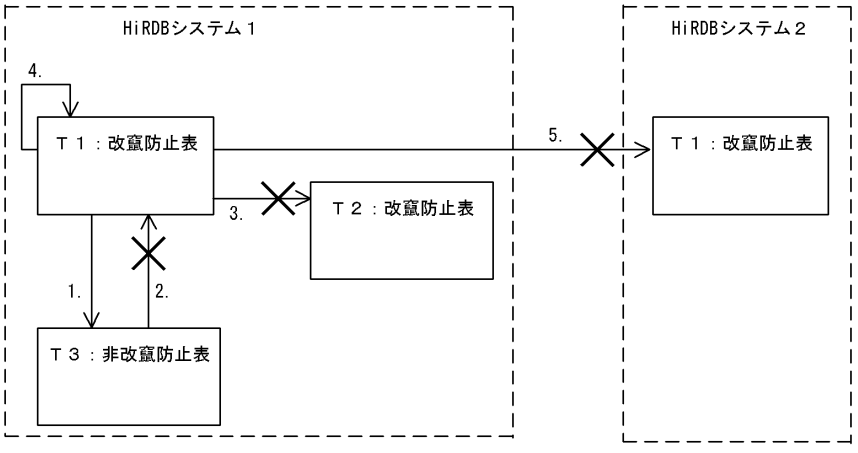
**表 12-7 運用時に制限があるユティリティと制限事項**

| ユティリティ名 | 制限事項 |
|---|---|
| データベース作成ユティリティ（pdload） | • 作成モード（-d オプション指定）は使用できません。<br>• 対象となる表がデータ未完状態※の場合は実行できません。 |
| データベース構成変更ユティリティ（pdmod） | • 改竄防止表格納 RD エリアの再初期化（initialize rdarea）は実行できません。 |
| データベース再編成ユティリティ（pdrorg） | **表の再編成（-k rorg）**<br>• 処理対象となる表格納 RD エリアがコマンド閉塞となっていない場合は実行できません。<br>• UOC を利用した再編成（unlduoc 文）はできません。<br>• 同期点指定の再編成（option job 文）はできません。<br>• 対象となる表がデータ未完状態※の場合は実行できません。<br>**表のアンロード（-k unld）**<br>• -W オプションを指定しない場合は実行できません。<br>**表のリロード（-k reld）**<br>• 処理対象となる表格納 RD エリアがコマンド閉塞となっていない場合は実行できません。<br>• 対象となる表がデータ未完状態※となっている場合に実行できます（表の再編成で，表のリロードが異常終了した場合での再実行だけ実行できます）<br>• 同期点指定の再編成（option job 文）はできません。<br>• 別表へのリロードは実行できません。詳細は図 12-23 を参照してください。<br>**インデクスの一括作成（-k ixmk），再作成（-k ixrc），再編成（-k ixor）**<br>対象となる表がデータ未完状態※の場合は実行できません。 |
| リバランスユティリティ（pdrbal） | 対象となる表がデータ未完状態※の場合は実行できません。 |

注※

改竄防止表に対して表の再編成を実行し，エラーなどによって表のリロードが完了していない表の状態を**データ未完状態**といい，改竄防止表格納 RD エリアごとに状態を保持します。該当する RD エリアがデータ未完状態かどうかは，データベース状態解析ユティリティで，RD エリア単位の状態解析（論理的解析）又は表単位の状態解析を実行することで確認できます。データ未完状態は表の再編成（表のリロード）が正常に完了した場合に解除されます。データ未完状態の解除方法の詳細は，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

**図 12-23 別表へのリロード可否**




〔説明〕

1. 改竄防止表 T1 から非改竄防止表 T3 へのリロードは，改竄防止表 T1 の改竄にはならないため，できます。
2. 非改竄防止表 T3 から改竄防止表 T1 へのリロードは，改竄防止表 T1 のデータが改竄されることになるため，できません。
3. 改竄防止表 T1 から改竄防止表 T2 へのリロードは，改竄防止表 T2 のデータが改竄されることになるため，できません。
4. 改竄防止表 T1 自身に対するリロードは，改竄防止表 T1 の改竄にはならないため，できます。
5. HiRDB システム 1 の改竄防止表 T1 から HiRDB システム 2 の改竄防止表 T1 へのリロードは，HiRDB システム 2 の改竄防止表 T1 のデータが改竄されることになるため，できません。

## (3) 運用コマンド

改竄防止表，又は改竄防止表格納 RD エリアに対して，コマンド実行が制限されます。制限事項がある運用コマンドを次の表に示します。

**表 12-8 制限がある運用コマンドと制限事項**

| 運用コマンド | 制限事項 |
|---|---|
| RD エリアの閉塞（pdhold） | データ未完状態の改竄防止表格納 RD エリアの場合，状態を遷移させると表の再編成を完了させるためのリロード実行が不可となるため次のオプションは実行できません。<br>• 参照可能閉塞：-i<br>• バックアップ閉塞：-b |
| RD エリアの閉塞解除（pdrels） | データ未完状態の改竄防止表格納 RD エリアの場合，表の再編成が未完了のままで閉塞解除すると該当表のデータが 0 件の状態となるため，実行できません。 |

## (4) 関連製品での制限事項

関連製品での制限事項を次に示します。

- レプリケーション機能
  改竄防止表に対して，レプリケーション機能（HiRDB Dataextractor 及び HiRDB Datareplicator）を使用してデータの複写，及び更新結果を反映しないでください。使用すると，反映元と反映先でデータの内容が不一致となったり，エラーとなる場合があります。

## 12.15.3 非改竄防止表から改竄防止表への変更

既存の表を改竄防止表に変更する方法について説明します。次に示すどちらかの方法で，表を改竄防止表に変更できます。

- HiRDB Control Manager の改竄防止ウィザードを使用する方法
- HiRDB のコマンドを使用する方法

なお，データが格納されている表は改竄防止表に変更できません。したがって，表にデータが格納されている場合は，いったん表データをアンロードし，その後に ALTER TABLE で表の定義を変更します。

### (1) HiRDB Control Manager の改竄防止ウィザードを使用する方法

HiRDB Control Manager の改竄防止ウィザードを使用して，改竄防止表に変更する手順を次に示します。なお，手順の画面は Windows 版 HiRDB サーバで実行した例です。UNIX 版 HiRDB サーバで実行する場合は，パス名の表記が異なります。

**〈手順〉**

表 T1 を次に示す条件を設定した改竄防止表に変更します。

- COL_NOTE 列は更新可能列にします。
- COL_DATE 列を挿入履歴保持列とし，行削除禁止期間を 10 年に設定します。

1. HiRDB Control Manager - Console を起動します。HiRDB Control Manager - Console の起動方法については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。
2. 操作対象の HiRDB サーバを登録します。管理 HiRDB の登録方法については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。既に登録してある場合，この手順は必要ありません。
3. タブメニュー［表操作］から［改ざん防止表移行ウィザード］を選択します。
［改ざん防止ウィザード - 1/5］画面が表示されます。

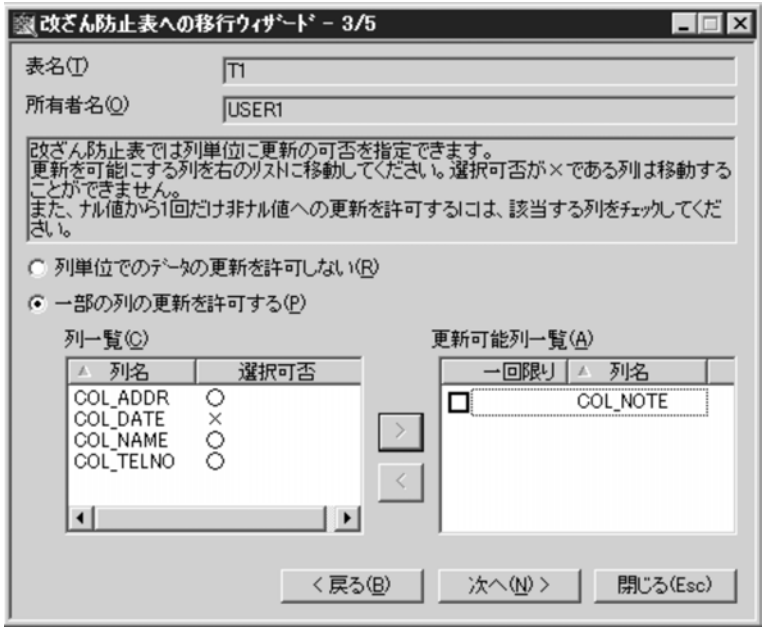
改竄防止表に変更する表名を選択して，［パスワード］テキストボックスに，表所有者のパスワー

ドを入力してください。ここで入力する値は，大文字と小文字の区別をします。

4. ［次へ>］ボタンをクリックします。
   ［改ざん防止ウィザード - 2/5］画面が表示されます。

この画面では，行削除禁止期間を設定します。行削除禁止期間を設定しない場合は，［表データの削除を行わない］が選択されていることを確認してください。行削除禁止期間を設定する場合は，［一定期間後に表のデータの削除を許可する］を選択し，［挿入履歴保持列名］と［行削除禁止期間］を設定してください。

5. ［次へ>］ボタンをクリックします。
   ［改ざん防止ウィザード - 3/5］画面が表示されます。




この画面では，更新可能列を設定します。更新可能列を設定しない場合は，［列単位でのデータの更新を許可しない］が選択されていることを確認してください。更新可能列を設定する場合は，［一部の列の更新を許可する］を選択してください。次に，［列一覧］リストから更新可能にする列名を選択し，［>］ボタンをクリックして［更新可能列一覧］リストに追加してください。

6. ［次へ>］ボタンをクリックします。
［改ざん防止ウィザード - 4/5］画面が表示されます。

この画面では，変更処理中に使用する一時ファイルを格納するディレクトリについて指定します。

［一時ファイル格納先一覧］リストに既定値が設定されていますので，必要に応じて変更してください。

7. ［次へ>］ボタンをクリックします。
［改ざん防止ウィザード - 5/5］画面が表示されます。

設定した条件を確認します。設定を変更したい場合は，［<戻る］ボタンをクリックして，前の画面に戻ります。

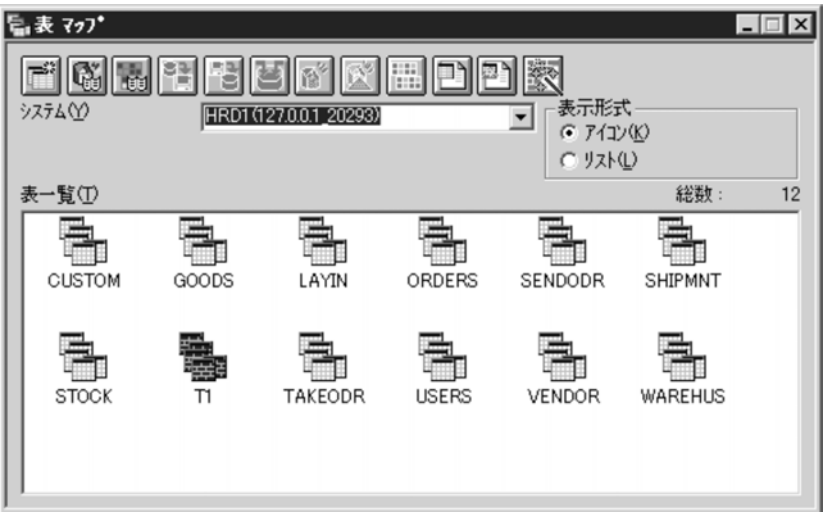
8. 問題がなければ［実行］ボタンをクリックします。変更処理が始まります。

**! 注意事項**

改竄防止ウィザード実行後は，ディクショナリ用 RD エリアと表データの回復に必要な RD エリアのバックアップを取得してください。バックアップは，HiRDB Control Manager のバックアップウィザードで実行できます。

**参考**

HiRDB Control Manager では，改竄防止表は通常表と区別して表示されます。タブメニュー［マップ］から［表］を選択して，［表マップ］画面を表示すると，変更した表が改竄防止属性であることが確認できます。




## (2) HiRDB のコマンドを使用する方法

HiRDB のコマンドを使用して，改竄防止表に変更する手順を次に示します。

**〈手順〉**

RD エリア（RDAREA01）に格納されている表 T1 を改竄防止表に変更します。

1. pdhold コマンドで，非改竄防止表が格納されている RD エリアとデータディクショナリ用 RD エリア（RDDIC01）をバックアップ閉塞します。

```
pdhold -r RDAREA01,RDDIC01 -b
```

2. バックアップ対象の RD エリアが属するサーバ（bes01，dic01）のシステムログファイルをスワップさせます。

```
pdlogswap -d sys -s bes01 -w
pdlogswap -d sys -s dic01 -w
```

3. データベース複写ユティリティ（pdcopy）を実行して RD エリアのバックアップを取得します。バックアップの取得方法については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

```
pdcopy -m C:¥hirdb¥rdarea¥mast¥mast01 -M r -p
C:¥usr¥hirdb¥pdcopy¥pdcopy01 -b C:¥usr¥hirdb¥pdcopy¥backup¥backup01 -r
RDAREA01,RDDIC01
```

4. pdrels コマンドで，データディクショナリ用 RD エリアの閉塞を解除します。

```
pdrels -r RDDIC01
```

5. 非改竄防止表のデータを，データベース再編成ユティリティ（pdrorg）を使用してアンロードします。このとき，後でデータベース作成ユティリティ（pdload）の入力データとして使用できるように，-W オプションを指定してアンロードしてください。制御文ファイル（control_file）については，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

```
pdrorg -k unld -t T1 -W bin control_file
```

6. pdrels コマンドで，ユーザ用 RD エリアの閉塞を解除します。なお，9. で再度 RD エリアを閉塞するまで，これ以降 RD エリアへのアクセスは行わないでください。この間に対象となる表が更新された場合，データの不整合が発生するおそれがあります。

```
pdrels -r RDAREA01
```

7. 非改竄防止表のデータを PURGE TABLE ですべて削除します。

```
PURGE TABLE T1
```

8. ALTER TABLE で改竄防止オプションを指定して，改竄防止表に変更します。

```
ALTER TABLE T1 CHANGE INSERT ONLY
```

9. pdhold コマンドで，改竄防止表が格納されている RD エリアを閉塞します。

```
pdhold -r RDAREA01
```

10. 5. でアンロードしたデータを，データベース作成ユティリティ（pdload）でロードします。制御文ファイル（control_file）については，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

```
pdload -b -W T1 control_file
```

11. pdrels コマンドで，RD エリアの閉塞を解除します。

```
pdrels -r RDAREA01
```

12. pdhold コマンドで，バックアップ対象の RD エリアをバックアップ閉塞します。

```
pdhold -r RDAREA01,RDDIC01 -b
```

13. バックアップ対象の RD エリアが属するサーバ（bes01，dic01）のシステムログファイルをスワップさせます。

```
pdlogswap -d sys -s bes01 -w
pdlogswap -d sys -s dic01 -w
```

14. データベース複写ユティリティ（pdcopy）を実行して RD エリアのバックアップを取得します。バックアップの取得方法については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

```
pdcopy -m C:¥hirdb¥rdarea¥mast¥mast01 -M r -p
C:¥usr¥hirdb¥pdcopy¥pdcopy01 -b C:¥usr¥hirdb¥pdcopy¥backup¥backup01 -r
RDAREA01,RDDIC01
```

15. pdrels コマンドで，RD エリアの閉塞を解除します。

```
pdrels -r RDAREA01,RDDIC01
```

なお，改竄防止オプションの設定時期はディクショナリ表の SQL_TABLES 表の値で確認できます。SQL_TABLES 表の値の意味を次の表に示します。

**表 12-9　SQL_TABLES 表の値の意味**

| 改竄防止オプションの設定 | SQL_TABLES | |
|---|---|---|
| | INSERT_ONLY 列の値 | CHANGE_TIME_INSERT_ONLY の値 |
| なし | NULL | NULL |
| CREATE TABLE 実行時に指定 | Y | NULL |
| ALTER TABLE 実行時に指定 | Y | 改竄防止表への変更日時 |

## 12.15.4　障害時の運用

改竄防止表格納 RD エリアは再初期化（initialize rdarea）できないため，RD エリアの回復のとき，再初期化を使った回復はできません。データベース回復ユティリティ（pdrstr）で回復してください。また，RD エリア満杯時は expand rdarea 文で RD エリアを拡張してください。

# 12.16　繰返し列を含む表

HiRDB では，複数の要素から構成される列（**繰返し列**）を含む表を定義できます。要素とは，繰返し列中で繰り返されている各項目のことをいいます。従来は，このような表を定義する場合，次の図のように作成する必要がありました。繰返し列を定義しない表の例を次の図に示します。

**図 12-24　繰返し列を定義しない表の例**

社員表

| 氏名 | 性別 | 資格 |
|---|---|---|
| 伊藤栄一 | 男 | 情報処理１種 |
| 伊藤栄一 | 男 | ネットワーク |
| 伊藤栄一 | 男 | 情報処理２種 |
| 中村和男 | 男 | 情報処理２種 |
| 中村和男 | 男 | 英語検定２級 |
| 河原秀雄 | 男 | シスアド |
| 井上俊夫 | 男 | |

扶養家族表

| 氏名 | 家族 | 続柄 | 扶養 |
|---|---|---|---|
| 伊藤栄一 | 虎夫 | 父 | 1 |
| 伊藤栄一 | ウメ | 母 | 1 |
| 伊藤栄一 | 綾子 | 妻 | 1 |
| 伊藤栄一 | 太郎 | 長男 | 1 |
| 伊藤栄一 | 恵子 | 次女 | 1 |
| 中村和男 | 和彦 | 父 | 0 |
| 中村和男 | 陽子 | 妻 | 1 |
| 河原秀雄 | 直子 | 母 | 1 |

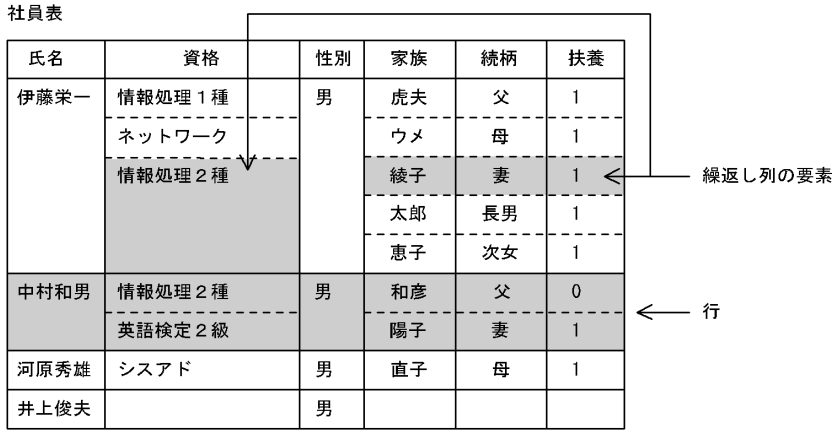
この二つの表をアクセスする場合，結合する必要があります。結合することで，SQL の構文が複雑になるなどのデメリットが発生します。そこで，繰返し列を含む表を作成することで，一つの表として作成できるため，結合が不要になります。

繰返し列を含む表の例を次の図に示します。

**図 12-25　繰返し列を含む表の例**

社員表

| 氏名 | 資格 | 性別 | 家族 | 続柄 | 扶養 |
|---|---|---|---|---|---|
| 伊藤栄一 | 情報処理１種 | 男 | 虎夫 | 父 | 1 |
| | ネットワーク | | ウメ | 母 | 1 |
| | 情報処理２種 | | 綾子 | 妻 | 1 |
| | | | 太郎 | 長男 | 1 |
| | | | 恵子 | 次女 | 1 |
| 中村和男 | 情報処理２種 | 男 | 和彦 | 父 | 0 |
| | 英語検定２級 | | 陽子 | 妻 | 1 |
| 河原秀雄 | シスアド | 男 | 直子 | 母 | 1 |
| 井上俊夫 | | 男 | | | |




注　空白の箇所は，ナル値を表します。

〔説明〕

「資格」，「家族」，「続柄」，「扶養」が繰返し列になります。

## (1) 繰返し列を含む表を定義したときの効果

多値多重性のある表を行ごとにまとめた形で表現できます。このため，次に示す効果が期待できます。

- 複数の表の結合が不要になります。
- 重複する情報がなくなるため，ディスク容量を削減できます。
- 関連データ（繰返しデータ）が近くに格納されるため，別の表にするよりアクセス性能が優れています。

### (2) 指定方法

繰返し列を指定するには，定義系 SQL の CREATE TABLE の列定義に ARRAY オプションを指定します。

繰返し列を含む表の定義例を次に示します。この定義例は図 12-25 に示した社員表の場合です。なお，社員表には，「続柄」，「扶養」に複数列インデクスが定義されているものとします。

（例）

```
CREATE TABLE 社員表
(氏名 NVARCHAR(10),
 資格 NVARCHAR(20) ARRAY[10],
 性別 NCHAR(1),
 家族 NVARCHAR(5) ARRAY[10],
 続柄 NVARCHAR(5) ARRAY[10],
 扶養 SMALLINT ARRAY[10]);

CREATE INDEX 扶養IDX ON 社員表 (続柄,扶養);
```

注　扶養 IDX は，社員表に付けたインデクス名です。

### (3) 注意

- 次に示すデータ型に対して繰返し列を指定できません。
  - BLOB 型
  - BINARY 型
  - 抽象データ型
- クラスタキーを指定した列には，繰返し列を指定できません。
- FIX を指定した場合，繰返し列を指定できません。
- 繰返し列に対して格納条件，ハッシュ分割，サプレスオプションを指定できません。
- キーレンジ分割をする場合，境界値を指定する列に繰返し列を指定できません。
- 繰返し列には非ナル値制約を指定できません。

# 12.17 抽象データ型を含む表

HiRDB では表を構成する列のデータ型として**抽象データ型**を定義できます。

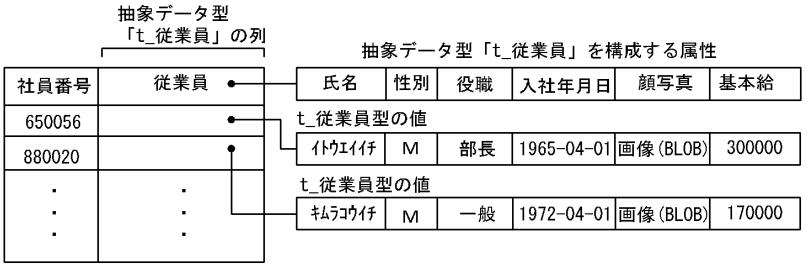
抽象データ型とは，既定義のデータ型で扱えないような複雑なデータを扱いやすく表現するための構造を持ったデータ型のことです。HiRDB では，このようなデータ型をユーザが抽象データ型として定義できます。抽象データ型では，構造を示す属性とその値に対する操作をひとまとまりとし，定義系 SQL によって定義できます。

抽象データ型は，数値型や文字型などの既定義のデータ型と同様に，表を構成する列のデータ型としてを扱えます。

抽象データ型を含む表のデータ構造を次の図に示します。次の図では，社員表中の「従業員」列のデータ型を抽象データ型「t_ 従業員型」としています。

**図 12-26 抽象データ型を含む表のデータ構造**

● 抽象データ型「t_従業員」を含む社員表




## (1) 抽象データ型を定義した場合の効果

- 複雑な構造のデータを一つの値として扱えます。
- データとそれに対する操作を一体化することで，オブジェクト指向のアプリケーションとマッピングしやすくなります。
- データとそのデータに対する操作をひとまとまりとし，操作を外部的なインタフェースとすることで，データの内部情報を意識することなくデータを扱えます。

## (2) 継承の概要

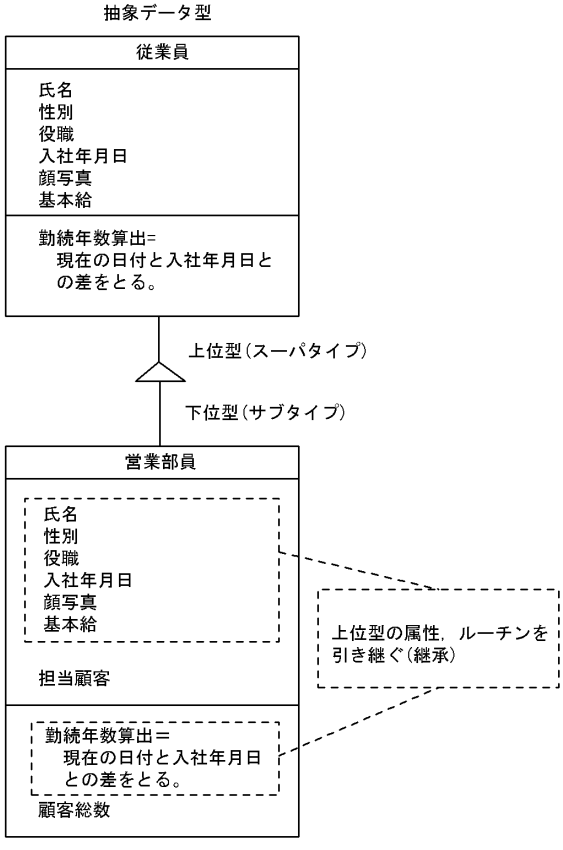
### (a) 継承（inheritance）

HiRDB では，既に定義した抽象データ型を基に，その型に定義された属性と操作を受け継いだ新しい抽象データ型を導出し定義できます。基になる型を**スーパタイプ**といい，導出した型を**サブタイプ**といいます。サブタイプがスーパタイプの属性及び関数を引き継ぐことを**継承**といいます。

スーパタイプ－サブタイプの関係は階層的に表現できます。これによって，複雑な概念モデルを抽象データ型を用いて階層化して表現できます。

抽象データ型のスーパタイプ－サブタイプ関係に基づく階層構造を次の図に示します。次の図では，抽象データ型「従業員」から，サブタイプ「営業部員」を導出しています。

**図 12-27　抽象データ型のスーパタイプ－サブタイプ関係に基づく階層構造**

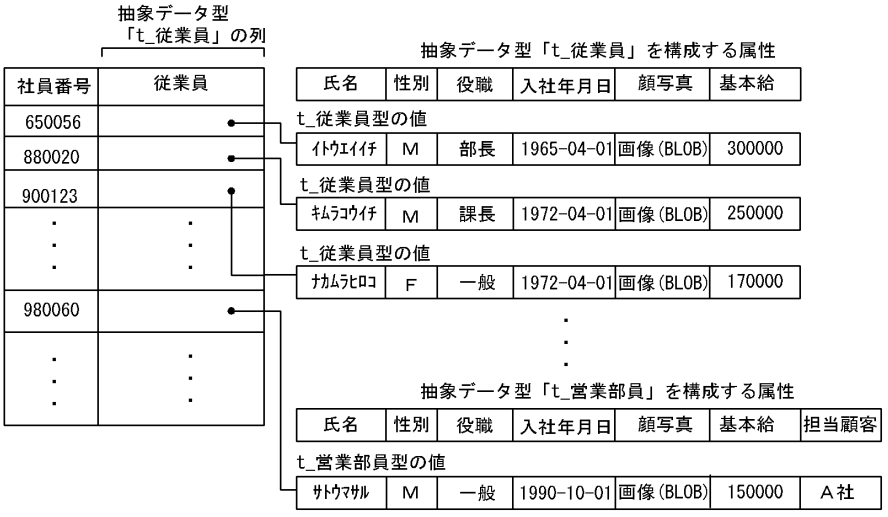



### (b) 代替可能性（substitutability）

サブタイプの値は，そのスーパタイプの値として扱うことができます。これを**代替可能性**（substitutability）といいます。代替可能性を利用して値を挿入した抽象データ型を含む表のデータ構造を次の図に示します。

**図 12-28 抽象データ型を含む表のデータ構造（代替可能性を利用した場合）**

● 抽象データ型「t_従業員」を含む社員表




### (c) 多重定義（override）

HiRDB では，上位の抽象データ型（スーパタイプ）で定義されたルーチンと同じ名前のルーチンを下位の抽象データ型（サブタイプ）の定義で上書きして定義できます。このように上書きして定義することを**多重定義**（override）といいます。多重定義によって，呼び出すルーチンの名称を型によって変更する必要がありません。

## (3) 継承を利用したときの効果

継承を利用することで，次に示す効果が期待できます。

- 上位の抽象データ型の特性（データ及び操作）を下位の抽象データ型でも利用できます。
- サブタイプを定義することで，最初から定義し直さなくてもデータ定義を共有できます。これによって，データベースの定義の手間が省けます。
- 多重定義によって，呼び出すルーチンの名称を型によって変更する必要がありません。

## (4) 抽象データ型の定義方法

抽象データ型を定義するには，定義系 SQL の CREATE TYPE を実行します。CREATE TYPE では，構造を示すための属性と，値に対する操作を定義します。また，継承を利用する場合は CREATE TYPE でサブタイプ句を指定して定義します。CREATE TYPE の定義例については，「6.5　ユーザが定義した抽象データ型を定義した表の作成」を参照してください。

### (a) コンストラクタ関数の定義

抽象データ型の値を生成するための関数（**コンストラクタ関数**）を定義します。コンストラクタ関数の実装で，抽象データ型の定義時に HiRDB によって提供される**デフォルトコンストラクタ関数**を利用できます。デフォルトコンストラクタ関数は，すべての属性がナル値である値を生成します。

### (b) ルーチンの定義

抽象データ型の定義内に，ある属性の値を操作するインタフェースとして**ルーチン**を定義できます。

### (c) 隠蔽レベルの指定

抽象データ型を構成する属性及びルーチンに対するアクセスを制御するため，**隠蔽レベル**を指定できます。隠蔽レベルは，属性及びその抽象データ型の値に対する操作であるルーチンに指定できます。隠蔽レベルには，次に示す 3 種類があります。

- PUBLIC
  その抽象データ型やサブタイプ以外の抽象データ型の定義中，アプリケーションからも属性の値へアクセスさせたい場合及びルーチンを使用させたい場合に指定します。
- PRIVATE
  内部情報がアプリケーションによって直接変更されることを防ぎたい場合などに，その抽象データ型の定義中だけで，属性の値へアクセスさせたい場合及びルーチンを使用させたい場合に指定します。SQL で属性の値へアクセスさせたい場合及びルーチンを使用したい場合は，関数を定義する必要があります。
- PROTECTED
  情報秘匿のため，アプリケーションから直接参照させたくない場合などに，その抽象データ型の定義中及びその抽象データ型のサブタイプの定義中でだけ属性の値へアクセスさせたい場合やルーチンを使用させたい場合に指定します。

抽象データ型の定義内でいったん隠蔽レベルを指定すると，次に別の隠蔽レベルの指定が出現するまでは直前の隠蔽レベルが有効になります。また，隠蔽レベルの指定がない場合，PUBLIC が仮定されます。隠蔽レベルの違いによって，データへのアクセス，ルーチンの使用権限の範囲が異なります。

隠蔽レベルと権限を次の表に示します。

**表 12-10 隠蔽レベルと権限**

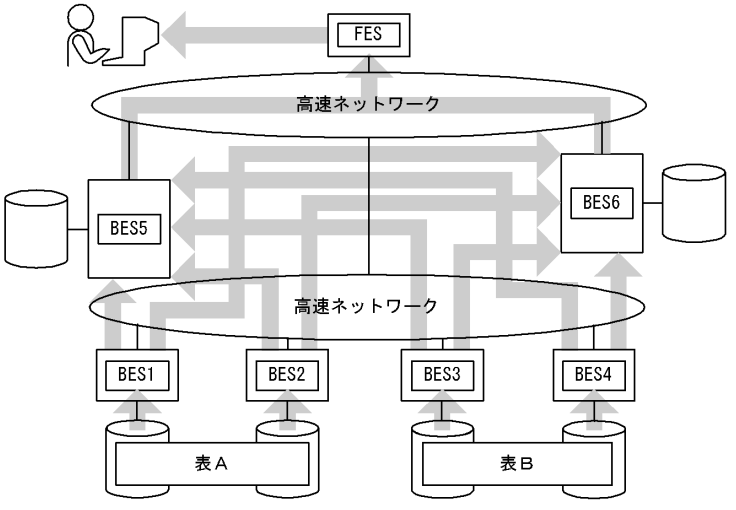
| 隠蔽レベル | アクセス元 | | | |
|---|---|---|---|---|
| | その抽象データ型の定義内 | サブタイプの抽象データ型の定義内 | 左記以外の抽象データ型の定義内 | アプリケーション |
| PUBLIC | ○ | ○ | ○ | ○ |
| PRIVATE | ○ | × | × | × |
| PROTECTED | ○ | ○ | × | × |

（凡例）
○：属性値へのアクセス及びルーチンを使用できます。
×：属性値へのアクセス及びルーチンを使用できません（SQL エラーになります）。

# 12.18　共用表

HiRDB/ パラレルサーバの場合，複数の表を結合するとき，それぞれの表が配置されたバックエンドサーバから表データを読み込み，別のバックエンドサーバで突き合わせ処理をします。そのため，複数のサーバを接続し，データを転送する処理が発生します。このとき，結合処理のための検索範囲のデータが一つのバックエンドサーバにあれば，そのデータを共用表として作成することで一つのバックエンドサーバで結合処理が完結します。**共用表**とは，**共用 RD エリア**に格納された表で，すべてのバックエンドサーバから参照できる表のことです。また，共用表に定義するインデクスを，**共用インデクス**といいます。共用表を更新できるのは**更新可能バックエンドサーバ**だけです。ほかのバックエンドサーバは**参照専用バックエンドサーバ**になります。ただし，共用表の更新には制限があるため，原則としてオンライン中は更新しないでください。共用表の更新については，「12.18.3　共用表の操作」を参照してください。共用表を使用しない結合処理を図 12-29 に，使用した結合処理を図 12-30 に示します。

**図 12-29　共用表を使用しない結合処理**




〔説明〕

表 A と表 B の結合処理をします。

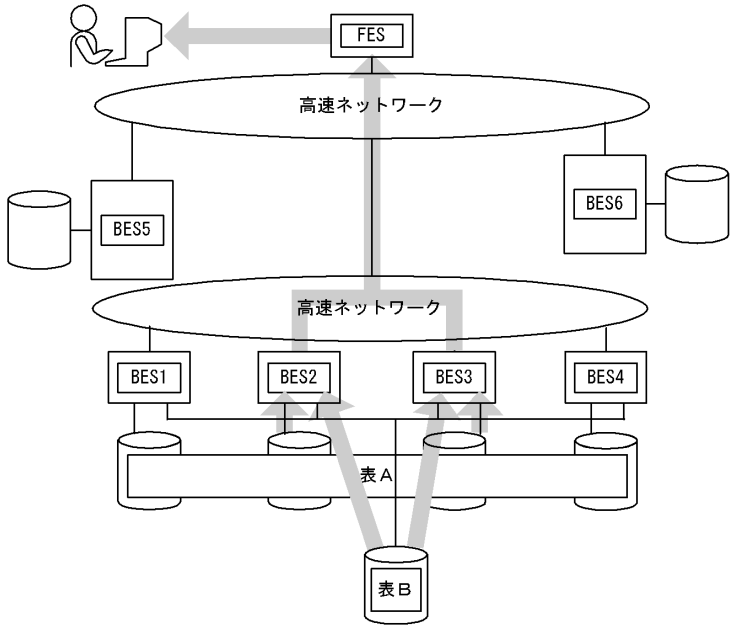
BES1，2：表 A からデータを取り出し，突き合わせ処理のため，BES5 と BES6 にデータを転送します。

BES3，4：表 B からデータを取り出し，突き合わせ処理のため，BES5 と BES6 にデータを転送します。

BES5，6：突き合わせ，結合処理をして FES にデータを転送します。

FES：結合処理した結果をマージして，ユーザに結果を返します。

**図 12-30　共用表を使用した結合処理**




〔説明〕

表 A と表 B の結合処理をします。共通データがある表 B を共用表とします。検索範囲は BES2，3 のバックエンドサーバにあります。

BES1，4，5，6：特に処理はありません。

BES2，3：表 A 及び表 B からデータを取り出し，マージ処理をして FES にデータを転送します。

FES：ユーザに結果を返します。

共用表及び共用インデクスは HiRDB/ シングルサーバにも定義できます。これによって，HiRDB/ パラレルサーバと SQL 及び UAP の互換性を保つことができます。共用表及び共用インデクスは HiRDB/ パラレルサーバで有効なので，通常は HiRDB/ パラレルサーバで使用します。ここでは，HiRDB/ パラレルサーバで共用表を使用する場合について説明します。HiRDB/ シングルサーバで共用表を使用する場合については，「12.18.6　HiRDB/ シングルサーバで共用表を使用する場合」を参照してください。

## 12.18.1　効果と適用基準

### (1) 共用表の効果

一つのバックエンドサーバで結合処理が完結するため，バックエンドサーバ間の接続やデータ転送によるオーバヘッドが削減できます。また，トランザクションごとに使用するバックエンドサーバ数を少なくできるため，多重実行時などに並列処理の効率が上がります。

### (2) 適用基準

更新処理が少なく，結合処理など複数のトランザクションから参照されるような表は，共用表として作成することをお勧めします。

## 12.18.2 定義方法

定義系 SQL の CREATE TABLE で SHARE を指定（CREATE SHARE FIX TABLE と指定）します。なお，共用表は次の条件を満たす必要があります。

- 共用表は非分割の FIX 表である
- 共用表，及び共用インデクスを格納する RD エリアは共用 RD エリア（pdfmkfs コマンドの -k オプションに SDB を指定）である
- WITHOUT ROLLBACK オプションが指定されていない
- 参照制約が定義された参照表でない

## 12.18.3 共用表の操作

### (1) 検索

共用表はすべてのバックエンドサーバから参照できるため，共用表を検索するために最適なバックエンドサーバを HiRDB が決定します。なお，共用表を更新すると全バックエンドサーバで排他が掛かるため，検索処理と更新処理とのデッドロックが発生することがあります。デッドロックを回避するために，共用表の検索時は次のようにすることをお勧めします。

- 排他オプションに WITHOUT LOCK，又は WITHOUT LOCK NOWAIT を指定する
- 共用表を更新するために検索する場合，FOR UPDATE 句を指定する

なお，共用表に対して IN EXCLUSIVE MODE 指定で LOCK 文を実行すると，対象の共用表と共用インデクスが格納されている RD エリアに排他を掛けます。検索する表が LOCK 文の対象ではなくても，アクセスする RD エリアが同じであれば排他制御されます。そのため，WITHOUT LOCK NOWAIT を指定していても，ほかのトランザクションが IN EXCLUSIVE MODE 指定で LOCK 文を実行している場合は，共用表にアクセスできません。そのため，IN EXCLUSIVE MODE 指定の LOCK 文実行中は共用表を検索できません。

### (2) 更新

共用表を更新する場合，LOCK 文で IN EXCLUSIVE MODE を指定し，全バックエンドサーバの共用 RD エリアに排他を掛けなければ実行できません。ただし，インデクスキー値を変更しない UPDATE 文は，LOCK 文を発行しないで実行できます。共用表及び共用インデクスの更新は，COMMIT 文発行時にディスクに書き込まれます。

なお，ローカルバッファを使用して共用表を更新する場合は，LOCK 文を発行して更新してください。LOCK 文を発行しない更新をしていて，サーバプロセスが異常終了すると，アボートコード Phb3008 が出力されます（ユニットが異常終了することがあります）。

#### (a) LOCK 文を発行する更新

LOCK 文を発行する更新の流れを次に示します。

1. IN EXCLUSIVE MODE 指定で LOCK 文を発行します。
   このとき，共用表だけでなく，共用表が格納されている共用 RD エリア及び共用インデクスが格納されている共用 RD エリアにも排他が掛かります。参照専用バックエンドサーバの共用 RD エリアのグローバルバッファは無効化されます。
2. 共用表に対して，INSERT，UPDATE，又は DELETE 文を実行します。
   更新可能バックエンドサーバが更新情報をファイルに反映します。
   LOCK 文が解除されるまで共用 RD エリアに排他が掛かるので，同じ共用 RD エリアにある，別の共

用表への操作は待ち状態になります。
3. LOCK 文を解除します。

**注意事項**

- LOCK 文は UAP の最初に発行してください。LOCK 文発行時，関連する共用 RD エリア内の表に対して，自サーバプロセスでオープン中のカーソルがあると，LOCK 文はエラーになります。
- 共用表を更新する手続き及びトリガを作成する場合，LOCK 文を記述してください。ただし，手続き及びトリガから LOCK 文を実行する場合はトランザクションの開始時点から排他が掛かりません。そのため，エラーになるおそれがあります。
- 全バックエンドサーバで共用表，共用表が格納されている共用 RD エリア，及び共用インデクスが格納されている共用 RD エリアに排他を掛けるため，該当する RD エリアの表又はインデクスにアクセスする業務があると，デッドロック，又はサーバ間のグローバルデッドロックが発生するおそれがあります。
- 共用表を更新するトランザクションの決着前に更新可能バックエンドサーバのユニットが異常終了して再開始しない場合に次のような検索をすると，排他タイムアウトエラーになります（KFPA11770-I メッセージが出力されます）。
  ・別のユニットの参照専用バックエンドサーバで，更新対象の共用表又はその表に定義されているインデクスが格納されている RD エリア内の表を検索

**(b) LOCK 文を発行しない更新**

LOCK 文を発行しない場合，実行できるのはインデクスキー値の変更がない UPDATE 文だけです。また，少量の変更の場合だけにしてください。

LOCK 文を発行しない更新の流れを次に示します。

1. 全バックエンドサーバの状態を同じにするため，更新情報を全バックエンドサーバに配布します。
2. 更新可能バックエンドサーバが更新情報をデータベースに反映します。
   参照専用バックエンドサーバでは，グローバルバッファ上で更新し，COMMIT 文発行までファイルに反映しないで更新情報を保持します。トランザクションがロールバックした場合，グローバルバッファ上で回復します。

**注意事項**

- 参照専用バックエンドサーバで，グローバルバッファがすべて更新中，かつ COMMIT 文未発行で空きページがない状態になった場合，トランザクションはロールバックします。このため，LOCK 文を発行しない場合は大量のデータを更新しないでください。
- 全バックエンドサーバで更新行に排他を掛けるため，同時に該当する表にアクセスする業務があるとデッドロック，又はサーバ間のグローバルデッドロックが発生するおそれがあります。デッドロックを回避するため，UPDATE 文で更新する行はトランザクションで 1 件だけにすることをお勧めします。
- 共用表を更新するトランザクションの決着前に更新可能バックエンドサーバのユニットが異常終了して再開始しない場合に次のような検索をすると，排他タイムアウトエラーになります（KFPA11770-I メッセージが出力されます）。
  ・別のユニットの参照専用バックエンドサーバで，更新対象の共用表又はその表に定義されているインデクスが格納されている RD エリア内の表を検索

## 12.18.4 共用表の制限事項

- IN EXCLUSIVE MODE 指定の LOCK 文実行中は共用表を検索できません。
- 共用表に対しては，ASSIGN LIST 文でリストを作成できません。
- レプリケーションの反映先に共用表は指定できません。

## 12.18.5 定義系 SQL，ユティリティ，及び運用コマンド実行時の注意

定義系 SQL，ユティリティ，及び運用コマンドで共用表又は共用インデクスを対象とする場合，HiRDB が内部的に LOCK 文を発行し，全バックエンドサーバで表と処理対象の RD エリアに対して IN EXCLUSIVE MODE 指定で排他を掛けることがあります。このため，該当する RD エリア内の表やインデクスにアクセス中の業務があると，デッドロック，又はサーバ間のグローバルデッドロックが発生するおそれがあります。

HiRDB が内部的に LOCK 文を発行する定義系 SQL を次に示します。

- 共用表に対する CREATE TABLE，DROP TABLE，PURGE TABLE
- 共用インデクスに対する CREATE INDEX，DROP INDEX
- 共用表を含むスキーマに対する DROP SCHEMA
- 共用表に対する空き領域再利用機能の変更（ALTER TABLE）

HiRDB が内部的に LOCK 文を発行するユティリティを次に示します。

- データベース作成ユティリティ（pdload）
- データベース再編成ユティリティ（pdrorg -k reld，rorg，ixrc，ixmk，ixor）
- 空きページ解放ユティリティ（pdreclaim）
- データベース定義ユティリティ（pddef）
- データディクショナリ搬出入ユティリティ（pdexp）
- データベース構成変更ユティリティ（pdmod -a initialize rdarea）

共用 RD エリアに対して実行できないユティリティや運用コマンドについては，「14.6(7) 共用 RD エリア使用上の制限事項」を参照してください。

## 12.18.6 HiRDB/ シングルサーバで共用表を使用する場合

HiRDB/ パラレルサーバの場合との相違点について説明します。

**注意事項について**

HiRDB/ シングルサーバで共用表を使用する場合の注意（共用表の操作時の注意事項，共用表の制限事項，及び定義系 SQL，ユティリティ，及び運用コマンド実行時の注意）は，基本的に HiRDB/ パラレルサーバと同じです。ただし，HiRDB/ シングルサーバはサーバが一つだけなので，サーバ間のデッドロックは発生しません。また，HiRDB/ パラレルサーバの場合の運用コマンド実行時の注意は，HiRDB/ シングルサーバの場合は該当しません。

**共用表及び共用インデクスの格納 RD エリアについて**

HiRDB/ シングルサーバでは共用 RD エリアを定義できないため，共用表及び共用インデクスは，通常のユーザ用 RD エリアに格納してください。このとき，共用表及び共用インデクスを格納するユーザ用 RD エリアと，共用表ではない表及び共用インデクスではないインデクスを格納するユーザ用 RD エリアは別にしてください。同じユーザ用 RD エリアに混在していると，デッドロックが発生するおそれがあります（共用表を更新中は，共用表及び共用インデクスが格納されている RD エリアにも排他が掛かるため，該当する RD エリアの表又はインデクスにアクセスする業務があると排他待ちになります）。

**ローカルバッファの使用について**

HiRDB/ シングルサーバでローカルバッファを使用して共用表及び共用インデクスを更新する場合，LOCK 文を発行しない更新をしていて，サーバプロセスが異常終了しても，アボートコード Phb3008 を出力して HiRDB が異常終了することはありません。

### HiRDB/シングルサーバから HiRDB/パラレルサーバへの移行について

移行する場合，HiRDB/ シングルサーバのシステム内に共用表及び共用インデクスが定義されている状態で，データベース構成変更ユティリティ（pdmod）を使用して HiRDB/ パラレルサーバに移行しないでください。移行手順を次に示します。

1. HiRDB/ シングルサーバで定義している共用表及び共用インデクスがあるかどうかを確認します。次に示す SQL 文（システム内に定義されている共用表の名称を確認するため，データディクショナリ表（SQL_TABLES）を検索する）を実行します。表名が出力されなければ，共用表は定義されていません。表名が出力されたら，共用表が定義されています。

```
SELECT TABLE_NAME
FROM MASTER.SQL_TABLES
WHERE SHARED='S'
WITHOUT LOCK NOWAIT
```

2. HiRDB/ シングルサーバで定義している共用表及び共用インデクスがあれば，すべて削除します。
3. データベース構成変更ユティリティ（pdmod）を使用して HiRDB/ パラレルサーバに移行します。
4. HiRDB/ パラレルサーバで共用 RD エリアを定義し，共用表及び共用インデクスを定義し直し，共用 RD エリアに格納します。

# 12.19 参照制約

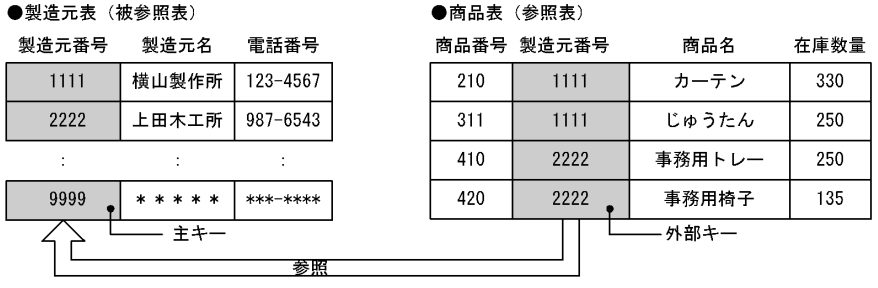
## 12.19.1 参照制約とは

データベース中の表は，それぞれ独立しているのではなく，お互いに関連を持っている場合があります。一方の表に関連するデータがないと，ほかの表でそのデータの意味がないことがあります。表間のデータの参照整合性を保つため，表定義時に特定の列（**外部キー**といいます）に定義する制約が**参照制約**です。参照制約及び外部キーを定義した表を**参照表**，外部キーによって参照表から参照される表を**被参照表**といいます。被参照表には外部キーによって参照される主キーを定義しておく必要があります。

なお，SQL やユティリティの実行などで被参照表と参照表間の参照整合性が保証できなくなる場合があります。この場合，参照表は検査保留状態になります。検査保留状態については「12.19.3　検査保留状態」を，整合性を保証できなくなる操作については「12.19.4　データ操作と整合性」を参照してください。

被参照表と参照表の例を次の図に示します。この例では，商品表が参照表，製造元表が被参照表となります。参照表の外部キーから主キーを参照し，製造元名が分かります。

**図 12-31　被参照表と参照表の例**




参照制約を定義する場合，外部キーにインデクスを定義すると処理性能が向上します。ただし，被参照表の主キーを更新しない場合は，外部キー値の更新によるインデクス更新のためのオーバヘッドがあるので，更新時の性能が悪くなることがあります。

**参照制約の効果**

参照制約を定義すると，表間のデータの整合性チェック，及びデータ操作を自動化できるので UAP を作成するときの負荷を軽減できます。ただし，被参照表や参照表を更新する場合，データの整合性をチェックするため，チェックに掛かる処理時間が増加しますので，注意してください。次のような場合に，チェックに掛かる処理時間が増加します。

- 更新する列が被参照表の主キーの場合
  被参照表の主キーを参照する参照表の外部キーの数に比例して遅くなります。
- 更新する列が参照表の外部キーの場合
  更新する列を構成列とする外部キーの数に比例して遅くなります。

## 12.19.2 参照制約の定義

参照制約を有効にするためには，外部キーによって参照される主キーを被参照表に定義しておく必要があります。定義系 SQL の CREATE TABLE で被参照表に PRIMARY KEY（主キー）を指定します。また，検査保留状態を使用するには，pd_check_pending **オペランド**に USE を指定するか，又はオペランドの指定を省略します。

参照表には，FOREIGN KEY（外部キー）を指定し，FOREIGN KEY 句中に次の指定をします。

- 参照する列
- 被参照表
- 参照制約動作
  参照制約動作は被参照表に対する挿入，更新，又は削除時の動作を CASCADE，又は RESTRICT で指定します。

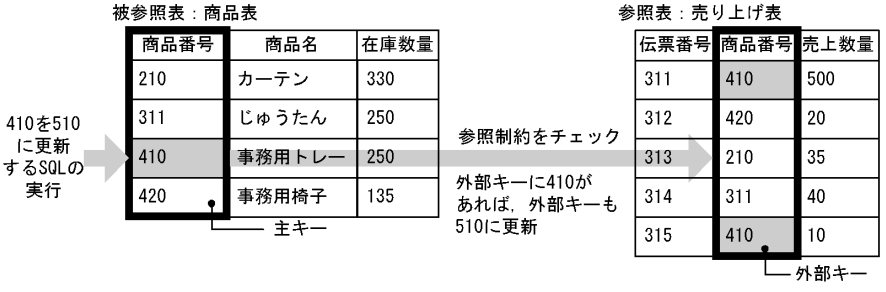
CASCADE，又は RESTRICT を指定した場合の被参照表と参照表の動作について説明します。

### (1) CASCADE を指定している場合

CASCADE を指定すると，被参照表の主キーに変更があった場合，外部キーも同じように変更されます。なお，参照表の外部キーに変更がある場合，主キーに変更後の値と同じ値の行があるかどうかをチェックして，参照制約違反エラーになれば外部キーは変更されません。

CASCADE を指定している場合，被参照表及び参照表に SQL を実行するときの動作の例を図 12-32 及び図 12-33 に示します。

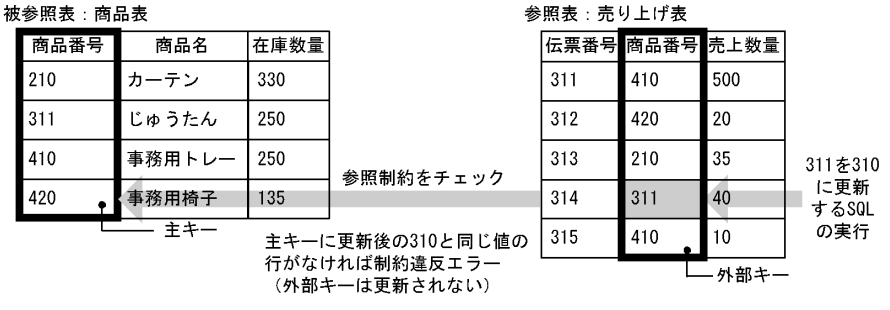
**図 12-32 被参照表に更新 SQL を実行するときの動作の例（CASCADE 指定時）**




〔説明〕

主キーの値と同じ値の行が外部キーにあれば，制約を保持するために，外部キーも主キーと同じように変更されます。この場合，被参照表への更新は実行されます。挿入及び削除も同じです。

**図 12-33 参照表に更新 SQL を実行するときの動作の例（CASCADE 指定時）**




〔説明〕

更新後の外部キーの値と同じ値の行が主キーにあれば，外部キーへの更新が実行されます。同じ値の行がなくても，外部キーにナル値があるときは外部キーへの更新が実行されます。ナル値がないときは参照制約違反エラーになります。この場合，被参照表には特に影響はありません。挿入及び削除も

同じです。

CASCADE を指定している場合の主キーに対する操作と参照表の動作を表 12-11 に，外部キーに対する操作と被参照表の動作を表 12-12 に示します。

**表 12-11　主キーに対する操作と参照表の動作（CASCADE 指定時）**

| 主キーに対する操作 | 被参照表と参照表の行の関係 | 主キーに対する操作結果 | 参照表の動作 |
|---|---|---|---|
| 挿入（INSERT 文） | なし | ○ | 動作しない |
| 更新（UPDATE 文），削除（DELETE 文） | 更新前の主キー構成列の値と同じ外部キー構成列の値を持つ行が，参照表にある | ○ | 主キーと同じ値で更新，又は行削除 |
| | 更新前の主キー構成列の値と同じ外部キー構成列の値を持つ行が，参照表にない | ○ | 動作しない |

（凡例）

○：正常に実行されます。

**表 12-12　外部キーに対する操作と被参照表の動作（CASCADE 指定時）**

| 外部キーに対する操作 | 参照表と被参照表の行の関係 | | 外部キーに対する操作結果 | 被参照表の動作 |
|---|---|---|---|---|
| 挿入（INSERT 文） | 挿入する行の外部キー構成列の値と同じ主キー構成列の値を持つ行が被参照表にある | | ○ | 動作しない |
| | 挿入する行の外部キー構成列の値と同じ主キー構成列の値を持つ行が被参照表にない | 外部キー構成列中にナル値がある | ○ | |
| | | 外部キー構成列中にナル値がない | × | |
| 更新（UPDATE 文） | 更新後の外部キー構成列の値と同じ主キー構成列の値を持つ行が被参照表にある | | ○ | 動作しない |
| | 更新後の外部キー構成列の値と同じ主キー構成列の値を持つ行が被参照表にない | 外部キー構成列中にナル値がある | ○ | |
| | | 外部キー構成列中にナル値がない | × | |
| 削除（DELETE 文） | なし | | ○ | 動作しない |

（凡例）

○：正常に実行されます。

×：制約違反エラーとなります。

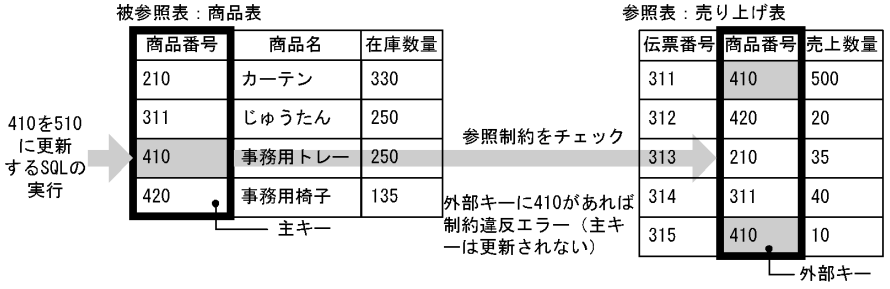
なお，CASCADE を指定すると，主キーの変更を外部キーにも反映するため，表定義時に HiRDB が内部的にトリガを生成します。参照制約動作のトリガ，及びユーザが定義するトリガとの関係については，「12.19.6　参照制約とトリガ」を参照してください。

### （2）RESTRICT を指定している場合

RESTRICT を指定すると，被参照表の主キーに変更がある場合，外部キーに同じ値の行があれば，参照制約違反エラーになり，主キーは変更されません。なお，外部キーに変更がある場合，主キーに同じ値の行があるかどうかをチェックして，参照制約違反エラーになれば外部キーは変更されません。

RESTRICT を指定している場合，被参照表に更新 SQL を実行するときの動作を次の図に示します。参照表の動作は，CASCADE 指定時（図 12-33 を参照）と同じです。

**図 12-34 被参照表に更新 SQL を実行するときの動作の例（RESTRICT 指定時）**




〔説明〕

主キーの値と同じ値の行が外部キーにあれば，参照制約違反エラーになり，主キーへの更新は実行されません。同じ値の行がなければ，被参照表への更新が実行されます。挿入及び削除も同じです。

RESTRICT を指定している場合の主キーに対する操作と被参照表及び参照表の動作を次の表に示します。外部キーに対する操作と被参照表の動作は CASCADE 指定時（表 12-12 を参照）と同じです。

**表 12-13 主キーに対する操作と被参照表及び参照表の動作**

| 主キーに対する操作 | 被参照表と参照表の行の関係 | 主キーに対する操作結果 | 参照表の動作 |
|---|---|---|---|
| 挿入（INSERT 文） | なし | ○ | 動作しない |
| 更新（UPDATE 文），削除（DELETE 文） | 更新前の主キー構成列の値と同じ外部キー構成列の値を持つ行が，参照表にある | × | 動作しない |
| | 更新前の主キー構成列の値と同じ外部キー構成列の値を持つ行が，参照表にない | ○ | |

（凡例）

○：正常に実行されます。

×：制約違反エラーとなります。

## (3) 被参照表，及び参照表定義時の制限事項

被参照表と参照表の表定義，表定義変更，及び表削除時の制限事項を次に示します。

### (a) 表定義（CREATE TABLE）時

- 同じ外部キー構成列（並びが同じでなくてもよい）の外部キーから，同じ被参照表を参照することはできません。
- 次の場合，外部キーは定義できません。
  - WITHOUT ROLLBACK を指定した表，共用表，及び改竄防止表の場合
  - WITHOUT ROLLBACK を指定した表に主キーが定義されていて，その主キーを参照する場合
- 外部キーは，一つの表に 255 個まで定義できます。256 個以上は定義できません。
- 一つの主キーに対して，外部キーは 255 個まで定義できます。256 個以上は定義できません。
- 参照表定義時に参照できる表は，同一スキーマの表だけです。
- 次の条件をすべて満たす場合にだけ，一つの表で，同じ主キーを参照する ON UPDATE CASCADE（更新時の参照制約動作が CASCADE）を指定した参照表を定義できます。
  - 複数の外部キー構成列が重複していない
  - 複数の外部キー構成列に関連する検査制約，又は参照制約を定義していない

- 外部キーの文字集合と，その外部キーから参照される表の主キーの文字集合は同じにしてください。

**(b) 表定義変更（ALTER TABLE 時）**

- 被参照表及び参照表に対して，DROP 句及び RENAME 句を使用した表定義変更はできません。
- 被参照表の主キー構成列，外部キー構成列に対して定義を変更する場合，次の制限があります。
  - CHANGE 句を使用したデータ型やデータ長の変更はできない
  - RENAME 句を使用した列名変更はできない
- WITH PROGRAM を指定した場合，参照表が参照する被参照表を使用する関数，手続き，及びトリガの SQL オブジェクトは無効になります。そのため，ALTER ROUTINE，ALTER PROCEDURE，又は ALTER TRIGGER で再作成する必要があります。

**(c) 表削除（DROP TABLE）時**

- 外部キーから参照される被参照表は削除できません。

## (4) 参照制約を定義する場合の注意事項

● 被参照表と参照表間のデッドロック

次の条件をすべて満たす場合，被参照表と参照表間でデッドロックが発生することがあります。これらの条件は参照制約動作が RESTRICT でも CASCADE でも同じです。

- 参照表の行を更新するトランザクションと，被参照表を更新するトランザクションが異なるトランザクションで，かつ同時に実行される
- 参照表で更新する行の主キー構成列の値と，被参照表で更新する行の外部キー構成列の値が同じである

被参照表及び参照表を操作する場合，上記条件が重ならないようにしてください。なお，それぞれのトランザクションで，操作対象の表に対して LOCK 文の排他モードで排他制御することでデータの整合性は保証できます。ただし，同時実行性は低下します。

● SQL オブジェクト用バッファ長の容量見積もり

参照制約動作を指定すると，HiRDB が内部的に制約条件チェックや参照制約動作を実行するトリガを生成するため，SQL オブジェクト用バッファを指定するときにそれらの SQL オブジェクトについても考慮する必要があります。SQL オブジェクト用バッファ長（pd_sql_object_cache_size）の見積もり式については，マニュアル「HiRDB Version 8 システム定義」を参照してください。

● データディクショナリ LOB 用 RD エリアの容量見積もり

参照制約動作に CASCADE を指定すると，参照制約動作を実行するトリガを HiRDB が生成します。このトリガのトリガ動作手続きの SQL オブジェクトはデータディクショナリ LOB 用 RD エリアに格納されます。そのため，参照制約動作に CASCADE を指定する場合はデータディクショナリ LOB 用 RD エリアに十分な容量を確保しておく必要があります。データディクショナリ LOB 用 RD エリアの容量の見積もりについては，「16.5　データディクショナリ LOB 用 RD エリアの容量の見積もり」を参照してください。

● バックアップの取得

バックアップは，被参照表が格納されている全 RD エリア，及び参照表が格納されている全 RD エリアの同期を合わせて取得してください。

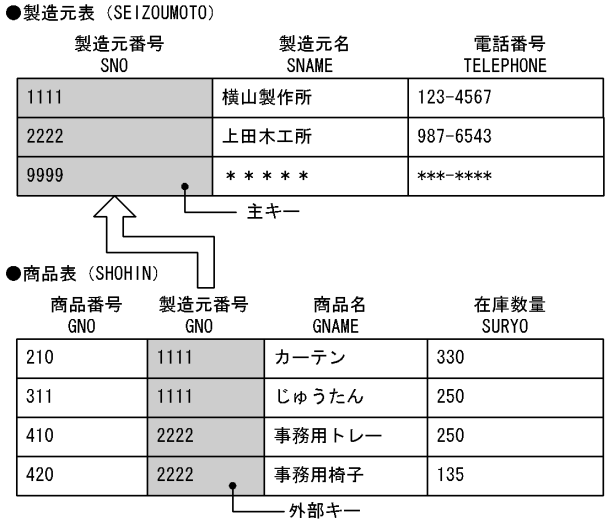
また，バックアップ取得時点の検査保留状態によって，バックアップの取得範囲が異なります。バックアップの取得時点と取得範囲については，マニュアル「HiRDB Version 8 システム運用ガイド」の「同時にバックアップを取得する必要がある RD エリア」を参照してください。

## (5) 参照制約の定義例

参照制約の定義例を次に示します。

### (a) 1対1対応で参照制約を定義する例

被参照表と参照表が 1 対 1 の場合の定義例を次に示します。




**参照制約の定義例1**

```
CREATE TABLE SEIZOUMOTO
  (SNO CHAR(4),SNAME NCHAR(6),TELEPHONE CHAR(12))
  PRIMARY KEY(SNO)  …主キーの指定
CREATE TABLE SHOHIN
  (GNO CHAR(4),SNO CHAR(4),GNAME NCHAR(10),SURYO INTEGER)
  CONSTRAINT SHOHIN_FK  …制約名の指定
  FOREIGN KEY(SNO)  …外部キーの指定
  REFERENCES SEIZOUMOTO  …被参照表名の指定
```

**参照制約動作の内容**

参照制約動作の指定を省略しているため，更新及び削除時は RESTRICT が仮定されます。製造元表の製造元番号（主キー）の更新，削除をする場合，商品表の製造元番号（外部キー）に対応する行があると，参照制約違反エラーとなり，製造元表の製造元番号の更新，削除は抑止されます。

**参照制約の定義例2**

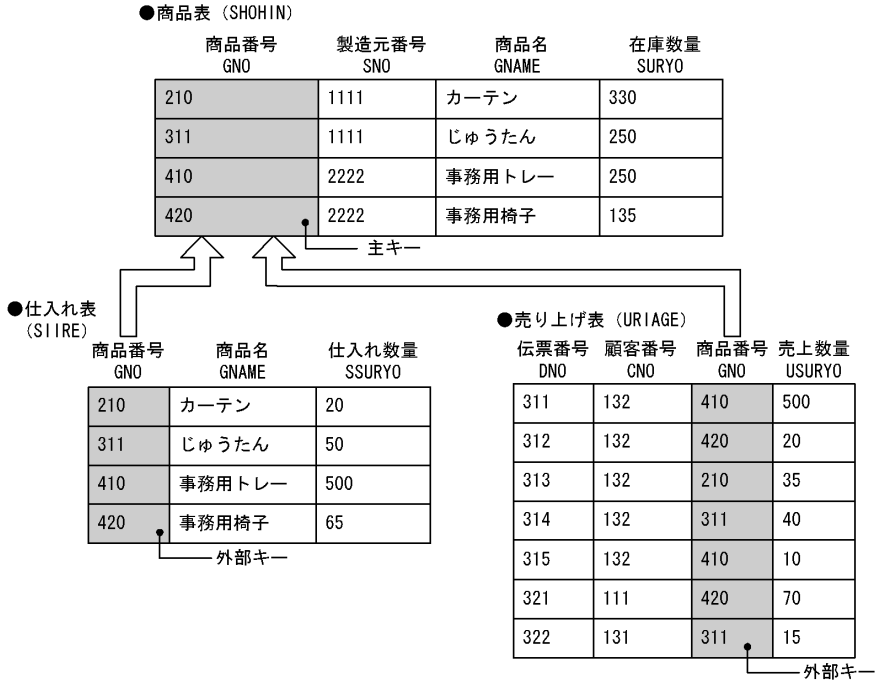
```
CREATE TABLE SEIZOUMOTO
  (SNO CHAR(4),SNAME NCHAR(6),TELEPHONE CHAR(12))
  PRIMARY KEY(SNO)  …主キーの指定
CREATE TABLE SHOHIN
  (GNO CHAR(4),SNO CHAR(4),GNAME NCHAR(10),SURYO INTEGER)
  CONSTRAINT SHOHIN_FK  …制約名の指定
  FOREIGN KEY(SNO)  …外部キーの指定
  REFERENCES SEIZOUMOTO  …被参照表名の指定
  ON UPDATE CASCADE  …更新時の参照制約動作の指定
  ON DELETE CASCADE  …削除時の参照制約動作の指定
```

**参照制約動作の内容**

製造元表の製造元番号（主キー）を更新した場合，対応する商品表の製造元番号（外部キー）も主キーと同じ値に更新されます。製造元表の行を削除した場合，商品表に対応する行も削除されます。

### (b) 1 対 2 対応で参照制約を定義する例

被参照表が 1，参照表が 2 の場合の定義例を次に示します。




**参照制約の定義例**

```
CREATE TABLE SHOHIN
  (GNO CHAR(4),SNO CHAR(4),GNAME NCHAR(10),SURYO INTEGER)
  PRIMARY KEY(GNO)  …主キーの指定
CREATE TABLE SIIRE
  (GNO CHAR(4),GNAME NCHAR(10),SSURYO INTEGER)
  CONSTRAINT SIIRE_FK  …制約名の指定
  FOREIGN KEY(GNO)  …外部キーの指定
  REFERENCES SHOHIN  …被参照表名の指定
  ON UPDATE CASCADE  …更新時の参照制約動作の指定
  ON DELETE CASCADE  …削除時の参照制約動作の指定
CREATE TABLE URIAGE
  (DNO CHAR(4),CNO CHAR(4),GNO CHAR(4),USURYO INTEGER)
  CONSTRAINT URIAGE_FK  …制約名の指定
  FOREIGN KEY(GNO)  …外部キーの指定
  REFERENCES SHOHIN  …被参照表名の指定
  ON UPDATE RESTRICT  …更新時の参照制約動作の指定
  ON DELETE RESTRICT  …削除時の参照制約動作の指定
```

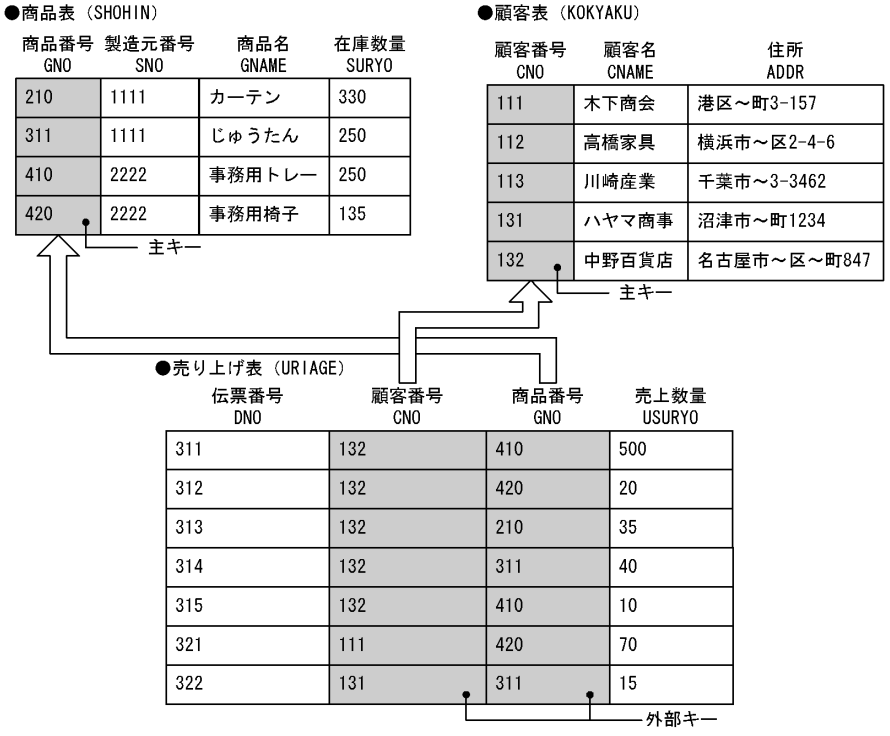
**参照制約動作の内容**

商品表の商品番号（主キー）を更新する場合，売り上げ表の商品番号（外部キー）に更新前の主キーと同じ値の行があると，参照制約違反エラーとなり，更新は抑止されます。売り上げ表に更新前の主キーと同じ値の行がないときは仕入れ表の対応する商品番号も主キーと同じ値に更新されます。

商品表の行を削除する場合，売り上げ表に更新前の主キーと同じ値の行があると，参照制約違反エラーとなり，削除は抑止されます。売り上げ表で更新前の主キーと同じ値の行がないときは仕入れ表の対応する行も削除されます。

### (c) 2対1対応で参照制約を定義する例

被参照表が2，参照表が1の場合の定義例を次に示します。




**参照制約の定義例**

```
CREATE TABLE SHOHIN
  (GNO CHAR(4),SNO CHAR(4),GNAME NCHAR(10),SURYO INTEGER)
  PRIMARY KEY(GNO)  …主キーの指定
CREATE TABLE KOKYAKU
  (CNO CHAR(4),CNAME NCHAR(8),ADDR NCHAR(24))
  PRIMARY KEY(CNO)  …主キーの指定
CREATE TABLE URIAGE
  (DNO CHAR(4),CNO CHAR(4),GNO CHAR(4),USURYO INTEGER)
  CONSTRAINT URIAGE_SHOHIN_FK  …制約名の指定
  FOREIGN KEY(GNO)  …外部キーの指定
  REFERENCES SHOHIN  …被参照表名の指定
  ON UPDATE CASCADE  …更新時の参照制約動作の指定
  ON DELETE CASCADE  …削除時の参照制約動作の指定
  CONSTRAINT URIAGE_KOKYAKU_FK
  FOREIGN KEY(CNO)  …外部キーの指定
  REFERENCES  KOKYAKU  …被参照表名の指定
  ON UPDATE CASCADE  …更新時の参照制約動作の指定
  ON DELEYE CASCADE  …削除時の参照制約動作の指定
```

**参照制約動作の内容**

商品表の商品番号（主キー）を更新する場合，売り上げ表の商品番号（外部キー）も同じ値に更新されます。商品表の行を削除する場合，売り上げ表の対応する行も削除されます。

顧客表の顧客番号（主キー）を更新する場合，売り上げ表の顧客番号（外部キー）も同じ値に更新されます。顧客表の行を削除する場合，売り上げ表の対応する行も削除されます。

## 12.19.3 検査保留状態

SQL やユティリティの実行などで表間の参照整合性を保証できなくなった場合，HiRDB は参照表に対するデータ操作を制限します。このように，整合性を保証できないためにデータ操作を制限された状態を**検査保留状態**といいます。参照表を検査保留状態にして，データ操作を制限するためには，pd_check_pending オペランドに USE を指定するか，又はオペランドの指定を省略する必要があります。検査保留状態の表は，整合性チェックユティリティ（pdconstck）を使用して検査保留状態を解除します。また，整合性チェックユティリティを使用して，強制的に検査保留状態にもできます。

pd_check_pending オペランドに NOUSE を指定していると，表間で参照整合性を保証できない場合でもデータ操作を制限しません。そのため，整合性が保証できなくなる SQL やユティリティを実行した場合は，整合性チェックユティリティで強制的に検査保留状態に設定してから，整合性を確認してください。

整合性が保証できなくなる操作については「12.19.4　データ操作と整合性」を，整合性の確認手順は「12.19.5　表の整合性確認手順」を参照してください。

### (1) 検査保留状態の設定又は解除

整合性チェックユティリティ以外に，次に示すユティリティ，コマンド，及び SQL で参照表を検査保留状態に設定するかどうかを決めたり，又は検査保留状態を解除したりできます。

- データベース作成ユティリティ（pdload）の constraint 文での指定
- データベース再編成ユティリティ（pdrorg）（リロード，再編成）の constraint 文での指定
- データベース構成変更ユティリティ（pdmod）（RD エリアの再初期化）
- PURGE TABLE 文
- ALTER TABLE（CHANGE RDAREA）

それぞれの詳細について，ユティリティ及びコマンドはマニュアル「HiRDB Version 8 コマンドリファレンス」を，SQL はマニュアル「HiRDB Version 8 SQL リファレンス」を参照してください。

### (2) 検査保留状態の管理

検査保留状態は**ディクショナリ表**と，表が格納された RD **エリアの表情報**で管理しています。ディクショナリ表では表単位，及び制約単位に検査保留状態を管理し，表情報では分割表の場合は RD エリア単位に，分割表ではない場合は表単位に検査保留状態を管理します。

検査保留状態の情報が格納されている場所と内容を次の表に示します。

**表 12-14　検査保留状態の情報が格納されている場所と内容（参照制約）**

| 格納場所 | | | 格納されている情報 |
|---|---|---|---|
| ディクショナリ表 | SQL_TABLES 表 | CHECK_PEND 列 | 表単位の参照制約の検査保留状態 |
| | SQL_REFERENTIAL_CONSTRAINTS 表 | CHECK_PEND 列 | 制約単位の参照制約の検査保留状態 |
| RD エリアの表情報 | | 分割していない表の場合 | 表単位の参照制約又は検査制約の検査保留状態 |
| | | 分割表の場合 | RD エリア単位の参照制約又は検査制約の検査保留状態 |

### (3) 検査保留状態の表に対して制限される操作

検査保留状態の表に対してできなくなる操作を次の表に示します。なお，トリガ動作によって操作対象表にアクセスする場合，トリガ SQL 文に指定した SQL の操作可否に依存します。また，操作対象表が

ビュー表の場合，ビュー表の基になる実表の操作可否に依存します。

**表 12-15 検査保留状態の表に対する操作可否**

| 検査保留状態の表に対する操作 | | | 操作可否 |
|---|---|---|---|
| 操作系 SQL | SELECT 文 | 対象表の検索 | △※1 |
| | | 対象表から作成したリストの検索 | |
| | INSERT 文 | 対象表への挿入 | |
| | UPDATE 文 | 対象表の更新 | |
| | DELETE 文 | 対象表からの行削除 | |
| | ASSIGN LIST 文 | 対象表からのリスト作成 | |
| ユティリティ | リバランスユティリティ（pdrbal） | | × |
| | データベース再編成ユティリティ（pdrorg） | 再編成 | △※2 |

（凡例）

△：場合によっては操作できません。

×：操作できません。

注※1

次の条件をどちらも満たす場合だけ，操作できます。それ以外は操作できません。

- 操作対象表が分割表で，分割条件がキーレンジ分割又は FIX ハッシュ分割
- 操作対象となる RD エリアが検査保留状態でない

注※2

フレキシブルハッシュ分割の分割表に対して再編成を実行する場合，操作できないときがあります。詳細は，マニュアル「HiRDB Version 8 コマンドリファレンス」の「データベース再編成ユティリティ（pdrorg）」の「規則及び注意事項」を参照してください。

## (4) 検査保留状態の表と参照関係がある表に対して制限される操作

次のような参照関係がある表を例に説明します。この場合，検査保留状態になるのは表 T2 と表 T3 だけです。

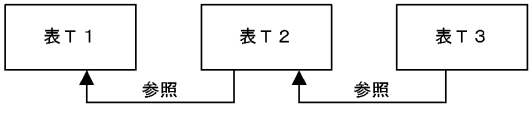



表 T2 と表 T3 が検査保留状態の場合，各表に対して制限される操作について次に示します。

### (a) 表 T2 だけが検査保留状態の場合

表 T2 だけが検査保留状態の場合，各表に対して制限される操作を次の表に示します。

**表 12-16 表 T2 だけが検査保留状態の場合，各表に対して制限される操作**

| 操作対象表 | 制限される操作 | 制限の内容 |
|---|---|---|
| 表 T1 | UPDATE（対象表の更新） | 表 T2 に定義されている参照制約動作指定によって異なります。<br>• CASCADE の場合<br>参照制約動作の操作対象となる RD エリアの表情報が検査保留状態のときは操作できません。ただし，同値更新のときは操作できます。<br>• RESTRICT の場合<br>操作できます。参照表 T2 を参照し，整合性チェックを行います。 |
| | DELETE（対象表からの行削除） | |
| 表 T2 | SELECT 文（対象表の検索及び対象表から作成したリストの検索） | 次の条件をどちらも満たす場合だけ，操作できます。それ以外は操作できません。<br>• 操作対象表が分割表で，分割条件がキーレンジ分割又は FIX ハッシュ分割<br>• 操作対象となる RD エリアが検査保留状態でない |
| | INSERT 文（対象表への挿入） | |
| | UPDATE 文（対象表の更新） | |
| | DELETE 文（対象表からの行削除） | |
| | ASSIGN LIST 文（対象表からのリスト作成） | |
| | リバランスユティリティ（pdrbal） | 操作できません。 |
| | データベース再編成ユティリティ（pdrorg）による再編成 | フレキシブルハッシュ分割の分割表に対して再編成を実行する場合，操作できないときがあります。詳細は，マニュアル「HiRDB Version 8 コマンドリファレンス」の「データベース再編成ユティリティ（pdrorg）」を参照してください。 |
| 表 T3 | 制限される操作はありません。INSERT 及び DELETE の場合，被参照表 T2 を参照し，整合性チェックを行います。 | |

### (b) 表 T3 だけが検査保留状態の場合

表 T3 だけが検査保留状態の場合，各表に対して制限される操作を次の表に示します。

**表 12-17 表 T3 だけが検査保留状態の場合，各表に対して制限される操作**

| 操作対象表 | 制限される操作 | 制限の内容 |
|---|---|---|
| 表 T1 | UPDATE（対象表の更新） | 表 T2 及び表 T3 に定義されている参照制約動作が CASCADE で，参照制約動作の操作対象となる RD エリアの表情報が検査保留状態のときは操作できません。ただし，同値更新のときは操作できます。 |
| | DELETE（対象表からの行削除） | |
| 表 T2 | UPDATE（対象表の更新） | 表 T2 及び表 T3 に定義されている参照制約動作指定によって異なります。<br>• CASCADE の場合<br>参照制約動作の操作対象となる RD エリアの表情報が検査保留状態のときは操作できません。ただし，同値更新のときは操作できます。<br>• RESTRICT の場合<br>操作できます。参照表 T3 を参照し，整合性チェックを行います。 |
| | DELETE（対象表からの行削除） | |

| 操作対象表 | 制限される操作 | 制限の内容 |
|---|---|---|
| 表 T3 | SELECT 文（対象表の検索及び対象表から作成したリストの検索） | 次の条件をどちらも満たす場合だけ，操作できます。それ以外は操作できません。<br>• 操作対象表が分割表で，分割条件がキーレンジ分割又は FIX ハッシュ分割<br>• 操作対象となる RD エリアが検査保留状態でない |
| | INSERT 文（対象表への挿入） | |
| | UPDATE 文（対象表の更新） | |
| | DELETE 文（対象表からの行削除） | |
| | ASSIGN LIST 文（対象表からのリスト作成） | |
| | リバランスユティリティ（pdrbal） | 操作できません。 |
| | データベース再編成ユティリティ（pdrorg）による再編成 | フレキシブルハッシュ分割の分割表に対して再編成を実行する場合，操作できないときがあります。詳細は，マニュアル「HiRDB Version 8 コマンドリファレンス」の「データベース再編成ユティリティ（pdrorg）」を参照してください。 |

### (c) 表 T2 及び表 T3 が検査保留状態の場合

表 T2 及び表 T3 が検査保留状態の場合，各表に対して制限される操作を次の表に示します。

**表 12-18 表 T2 及び表 T3 が検査保留状態の場合，各表に対して制限される操作**

| 操作対象表 | 制限される操作 | 制限の内容 |
|---|---|---|
| 表 T1 | UPDATE（対象表の更新） | 表 T2 及び表 T3 に定義されている参照制約動作が CASCADE で，参照制約動作の操作対象となる RD エリアの表情報が検査保留状態のときは操作できません。ただし，同値更新のときは操作できます。<br>表 T2 及び表 T3 に定義されている参照制約動作指定が RESTRICT の場合，操作できます。参照表 T2 を参照し，整合性チェックを行います。 |
| | DELETE（対象表からの行削除） | |
| 表 T2 | SELECT 文（対象表の検索及び対象表から作成したリストの検索） | 次の条件をどちらも満たす場合だけ，操作できます。それ以外は操作できません。<br>• 操作対象表が分割表で，分割条件がキーレンジ分割又は FIX ハッシュ分割<br>• 操作対象となる RD エリアが検査保留状態でない |
| | INSERT 文（対象表への挿入） | |
| | UPDATE 文（対象表の更新） | |
| | DELETE 文（対象表からの行削除） | |
| | ASSIGN LIST 文（対象表からのリスト作成） | |
| | リバランスユティリティ（pdrbal） | 操作できません。 |
| | データベース再編成ユティリティ（pdrorg）による再編成 | フレキシブルハッシュ分割の分割表に対して再編成を実行する場合，操作できないときがあります。詳細は，マニュアル「HiRDB Version 8 コマンドリファレンス」の「データベース再編成ユティリティ（pdrorg）」を参照してください。 |
| 表 T3 | SELECT 文（対象表の検索及び対象表から作成したリストの検索） | 次の条件をどちらも満たす場合だけ，操作できます。それ以外は操作できません。<br>• 操作対象表が分割表で，分割条件がキーレンジ分割又は FIX ハッシュ分割<br>• 操作対象となる RD エリアが検査保留状態でない |
| | INSERT 文（対象表への挿入） | |
| | UPDATE 文（対象表の更新） | |
| | DELETE 文（対象表からの行削除） | |
| | ASSIGN LIST 文（対象表からのリスト作成） | |

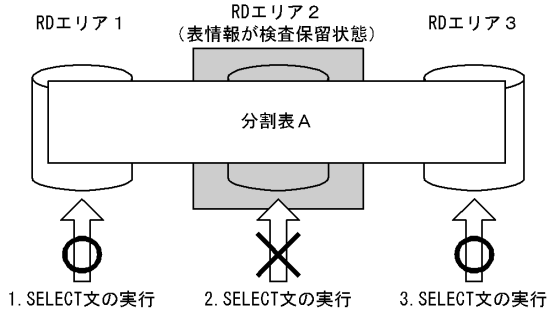
| 操作対象表 | 制限される操作 | 制限の内容 |
|---|---|---|
| | リバランスユティリティ（pdrbal） | 操作できません。 |
| | データベース再編成ユティリティ（pdrorg）による再編成 | フレキシブルハッシュ分割の分割表に対して再編成を実行する場合，操作できないときがあります。詳細は，マニュアル「HiRDB Version 8 コマンドリファレンス」の「データベース再編成ユティリティ（pdrorg）」を参照してください。 |

## (5) 分割表を使用している場合

RD エリア単位に検査保留状態を管理しているため，分割表で，実際に操作する RD エリアの表情報が検査保留状態の場合，その分割表に対する操作は制限されることがあります。それぞれについて説明します。

分割表で，データを格納している一部の RD エリアが検査保留状態の場合の例を次の図に示します。

**図 12-35 分割表で，RD エリア単位に検査保留状態を管理する場合のデータ操作可否**




〔説明〕

分割表 A に対して SELECT 文を実行する場合，実際に操作するデータが RD エリア 2（表情報が検査保留状態）にあると，SELECT 文はエラーになります。RD エリア 1 及び 3 にあるデータに対する操作の場合，SELECT 文は正常に実行できます。

**分割表の場合の注意事項**

pd_check_pending オペランドに USE を指定していて，参照表のデータを分割格納している RD エリアを再初期化する場合，再初期化後に整合性チェックユティリティで表単位の整合性チェックを実行してください。

## (6) 検査保留状態を使用する場合の注意

- pd_check_pending オペランドの指定値を NOUSE から USE に変更した場合，整合性チェックユティリティを使用して，参照表の整合性を確認する必要があります。確認手順については，「12.19.5 表の整合性確認手順」を参照してください。
- pd_check_pending オペランドに USE を指定していて，参照整合性を保証できなくなる操作をした場合でも，RD エリアの状態によっては検査保留状態を設定できないことがあります。このため，pd_check_pending オペランドの指定値を NOUSE から USE に変更すると，検査保留状態を使用していない場合は正常だった操作がエラーになることがあります。PURGE TABLE 文，又は ALTER TABLE（CHANGE RDAREA）実行時，検査保留状態を設定できる RD エリアの状態を次に示します。

  **オープン契機が INITIAL の場合**

  - RD エリアが閉塞なし，オープン状態のとき
  - RD エリアが更新可能バックアップ閉塞，かつオープン状態のとき

**オープン契機が DEFER 又は SCHEDULE の場合**

- RD エリアが閉塞なしのとき
- RD エリアが更新可能バックアップ閉塞のとき

ユティリティ実行時，検査保留状態を設定できる RD エリアについては，マニュアル「HiRDB Version 8 コマンドリファレンス」の「コマンド実行時の RD エリアの状態」の「検査保留状態の設定可否」を参照してください。

● pd_check_pending オペランドに USE を指定する場合，検査保留状態に設定される参照表及び RD エリアに対して排他が掛かるため，ユティリティ及び SQL 実行時の排他資源が検査保留状態を使用しない場合とは異なります。

## 12.19.4 データ操作と整合性

被参照表及び参照表に対する操作系 SQL（PURGE TABLE 文を除きます）による更新，追加，又は削除は，HiRDB が SQL 実行時にチェックし，整合性を保証します。ただし，表 12-19 及び表 12-20 に示す操作をした場合，整合性を保証できなくなることがあります。pd_check_pending オペランドに USE を指定していて，これらの操作をした場合，参照表は検査保留状態になります。

**表 12-19 整合性を保証できなくなる場合の，被参照表に対する操作とデータ不整合の発生条件**

| 表又は RD エリアに対する操作 | | データ不整合の発生条件 |
|---|---|---|
| データベース作成ユティリティ（pdload） | 作成モード（-d オプション指定）のデータロード | データロードした主キー構成列中に，参照表の外部キー構成列と同じ値を持つ行を含まない場合 |
| データベース再編成ユティリティ（pdrorg） | リロード（-k reld 指定） | リロードした主キー構成列中に，参照表の外部キー構成列と同じ値を持つ行を含まない場合 |
| | 再編成（-k rorg 指定） | UOC を使用して，参照表の外部キー構成列の値と同じ値の行を削除した場合 |
| データベース構成変更ユティリティ（pdmod） | RD エリアの再初期化（initialize rdarea） | 参照表が，再初期化した RD エリアと異なる RD エリアに格納されている場合 |
| PURGE TABLE 文 | | 参照表にデータが存在する場合 |
| ALTER TABLE による表の分割格納条件の変更 | | RD エリアの分割や統合の結果，参照表の外部キー構成列と同じ値を持つ行を含まない場合 |

**表 12-20 整合性を保証できなくなる場合の，参照表に対する操作とデータ不整合の発生条件**

| 表又は RD エリアに対する操作 | | データ不整合の発生条件 |
|---|---|---|
| データベース作成ユティリティ（pdload） | データロード | データロードした外部キー構成列中に，被参照表の主キー構成列と同じ値を持つ行がない場合 |
| データベース再編成ユティリティ（pdrorg） | リロード（-k reld 指定） | リロードした外部キー構成列中に，被参照表の主キー構成列と同じ値を持つ行がない場合 |

### (1) 操作対象の表が分割表の場合

操作対象の表が分割表で，表中に不整合データがある場合，ユティリティの実行によって，不整合データが，格納されている RD エリアを移動することがあります。例えば，RD エリア 1，2，及び 3 に分割格納されている表で，不整合データが RD エリア 1 にあるとき，ユティリティの実行によって，不整合データが RD エリア 3 に移動することがあります。不整合データの RD エリアの移動が発生する条件を次の表に示します。

表 12-21 操作対象が分割表の場合，表中の不整合データが RD エリアを移動する条件

| 表又は RD エリアに対する操作 | | 表中の不整合データが RD エリアを移動する条件 |
|---|---|---|
| データベース再編成ユティリティ（pdrorg） | 再編成（-k rorg 指定） | フレキシブルハッシュ分割表，又は第 2 次元分割列がフレキシブルハッシュ分割のマトリクス分割表に対して次の手順で操作する場合<br>1. RD エリア単位でデータロードをする<br>2. HiRDB/ シングルサーバの場合，表単位に再編成を実行する※<br>HiRDB/ パラレルサーバの場合，-g オプションを指定して表単位に再編成を実行する※ |
| リバランスユティリティ（pdrbal） | | 不整合データがある表に対して，RD エリアを追加してリバランスユティリティ（pdrbal）を実行する場合※ |

注※

pd_check_pending オペランドに USE を指定していて，操作対象表が検査保留状態の場合は実行できません。

### (2) データ不整合が発生する可能性があるそのほかの条件

次の条件をすべて満たす場合も，データ不整合が発生することがあるため，データの整合性を確認する必要があります。データの整合性確認手順については，「12.19.5　表の整合性確認手順」を参照してください。これらの条件は参照制約動作が RESTRICT でも CASCADE でも同じです。

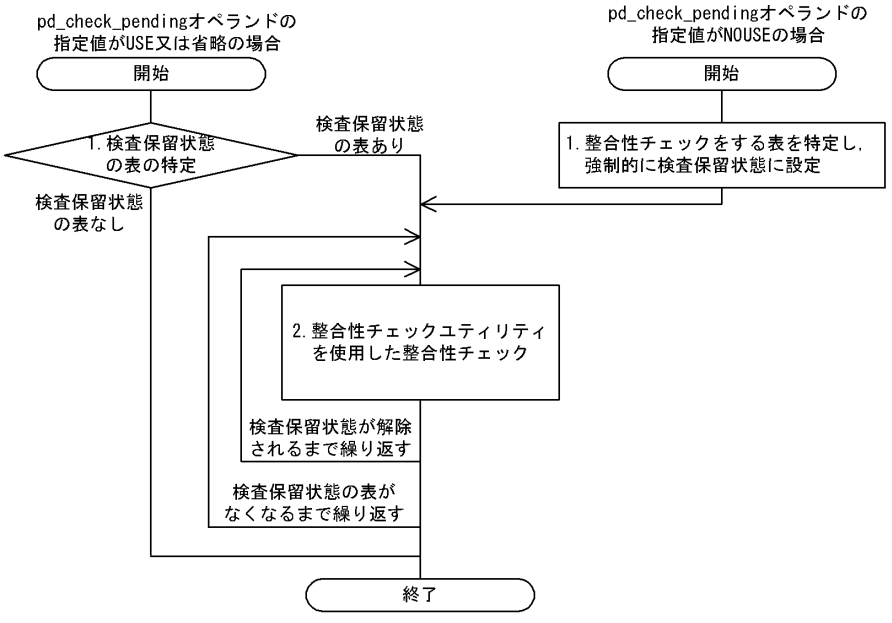
- 参照表の行を削除するトランザクションと，被参照表を更新又は削除するトランザクションが異なるトランザクションで，かつ同時に実行される
- 参照表で削除する行の主キー構成列の値と，被参照表で更新又は削除する行の外部キー構成列の値が同じである
- 参照表の行を更新又は削除するトランザクションをコミットし，被参照表の行を削除するトランザクションをロールバックする

被参照表及び参照表を操作する場合，上記条件が重ならないようにしてください。なお，それぞれのトランザクションで，操作対象の表に対して LOCK 文の共用モード又は排他モードで排他制御することでデータの整合性は保証できます。ただし，同時実行性は低下します。

## 12.19.5 表の整合性確認手順

データの整合性確認手順の概要を次の図に示します。

**図 12-36 データの整合性確認手順の概要（参照制約）**




pd_check_pending **オペランドの指定値が** USE **又は省略の場合**

1. **検査保留状態の表の特定**
   ディクショナリ表の SQL_TABLES 表を検索して，検査保留状態の表名を検出します。

```
SELECT TABLE_SCHEMA, TABLE_NAME FROM MASTER.SQL_TABLES
    WHERE CHECK_PEND = 'C' OR CHECK_PEND2 = 'C'
```

   検索結果には，検査保留状態の表の所有者と検査保留状態の表名が返されます。検索結果が 0 行の場合，検査保留状態の表はありません。

2. **整合性チェックユティリティを使用した整合性チェック**
   整合性チェックユティリティで表単位の整合性チェックを実行し，制約違反データがあれば修正します。検査保留状態の表がなくなるまで整合性チェックを繰り返し，なくなれば終了です。整合性チェックユティリティを使用した整合性確認手順については，「12.19.5(1) 検査保留状態を使用する場合の整合性確認手順（参照制約）」を参照してください。

pd_check_pending **オペランドの指定値が** NOUSE **の場合**

1. **整合性チェックをする表を特定し，強制的に検査保留状態に設定**
   整合性チェックをする表を特定するために，次のことを確認します。
   - 参照整合性を保証できなくなる操作をした表を被参照表とする参照表が存在するかどうか
   - 参照整合性を保証できなくなる操作をした表に参照制約が定義されているかどうか

   これらを確認する SQL の実行例を次に示します。

```
SELECT N_PARENTS, N_CHILDREN FROM MASTER.SQL_TABLES
    WHERE TABLE_SCHEMA = '対象表の所有者名' AND TABLE_NAME = '対象表の表名'
```

次の検索結果が返されます。
- 対象表に定義した外部キーの数
- 対象表に定義した主キーを参照する外部キーの数

N_PARENTS がナル値の場合，対象表に参照制約は定義されていません。
N_CHILDREN がナル値の場合，対象表を被参照表とする参照表は存在しません。
N_CHILDREN がナル値以外の場合，次に示す SQL を実行し，対象表を参照する参照表の表名を確認してください。

```
SELECT TABLE_SCHEMA, TABLE_NAME, CONSTRAINT_NAME
    FROM MASTER.SQL_REFERENTIAL_CONSTRAINTS
    WHERE R_OWNER = '対象表の所有者名' AND R_TABLE_NAME = '対象表の表名'
```

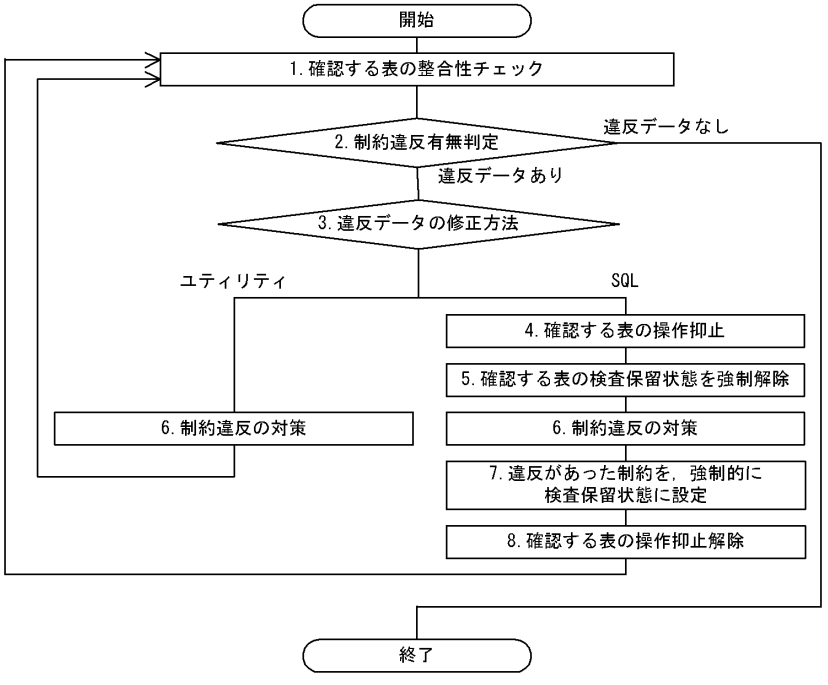
検索結果には，対象表を被参照表とする参照表の所有者名，表名及び参照制約の制約名が返されます。
検索結果が 0 行の場合，対象表を被参照表とする参照表はありません。
表を特定したら，整合性チェックユティリティを使用して，その表を強制的に検査保留状態に設定します（検査保留状態でない表は，整合性チェックユティリティでチェックできません）。

2. **整合性チェックユティリティを使用した整合性チェック**
pd_check_pending オペランドの指定値が USE 又は省略の場合の手順 2. と同じです。整合性チェックユティリティを使用した整合性確認手順については，「12.19.5(2) 検査保留状態を使用しない場合の整合性確認手順」を参照してください。

## (1) 検査保留状態を使用する場合の整合性確認手順（参照制約）

pd_check_pending オペランドの指定値が USE 又は省略の場合の，整合性チェックユティリティを使用した整合性確認手順を次の図に示します。

**図 12-37 検査保留状態を使用する場合の整合性確認手順（参照制約）**



1. **確認する表の整合性チェック**
   表単位，又は制約単位に整合性チェックをします。
2. **制約違反有無判定**
   手順 1. の整合性チェック結果で，制約違反データの有無を判定します。
3. **違反データの修正方法**
   違反データの修正をユティリティで行うか，SQL で行うかを選択します。ユティリティで行う場合，手順 6. へ進んでください。
4. **確認する表の操作抑止**
   整合性が保証できない表を使用する業務の運用を停止します。
5. **確認する表の検査保留状態を強制解除**
   制約違反の対策をするため，検査保留状態を強制解除します。
6. **制約違反の対策**

   **ユティリティで修正する場合**

   対策方法を次に示します。対策後，手順 1. に戻り，整合性チェックを実行し，違反データがないことを確認し，終了します。

| 条件 | 対策方法 |
|---|---|
| 主キーに必要なデータがない場合 | データベース作成ユティリティ（pdload）の追加モードで正しいデータをロードします。 |
| 外部キーに制約違反データがある場合 | • データベース作成ユティリティ（pdload）の作成モードで正しいデータをロードします。<br>• データベース再編成ユティリティ（pdrorg）の UOC を使用して不要なデータを削除します。 |

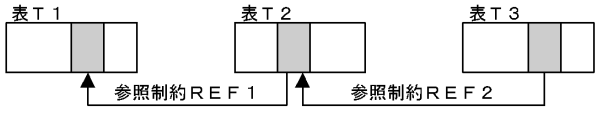
   **SQL で修正する場合**

   対策方法を次に示します。対策後，手順 7. に進みます。

| 条件 | 対策方法 |
|---|---|
| 主キーに必要なデータがない場合 | 主キーに必要なデータを INSERT 文で挿入します※1。又は，被参照表の既存のデータを UPDATE 文で更新します※2。 |
| 外部キーに制約違反データがある場合 | 外部キーの制約違反データを DELETE 文で削除，又は，UPDATE 文で正しい値に更新します※1。 |

注※1

外部キーが主キーでもあり，対策を行う表を被参照表とする参照表が存在する場合，修正順序に注意が必要です。例えば，次のような参照関係があるとします。




**● REF1 の制約違反の対策をする場合の注意**

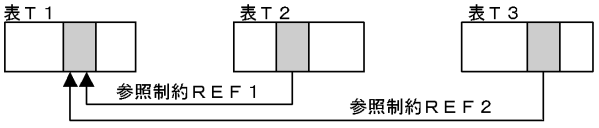
表 T2 のデータを DELETE 文で修正する場合，REF2 で ON DELETE RESTRICT を指定しているときは，対応する表 T3 のデータを先に削除後，表 T2 のデータを削除してください。また，UPDATE 文で修正する場合，REF2 で ON UPDATE RESTRICT を指定しているときは，更新前のデータに対応する表 T3 のデータを削除後，表 T2 のデータを更新してください。

**● REF2 の制約違反の対策をする場合の注意**

表 T2 のデータを INSERT 文で修正する場合，表 T1 に挿入対象のデータが存在するかどうか確認します。存在しないときは，先に表 T1 にデータを挿入後，表 T2 にデータを挿入してください。また，UPDATE 文で修正する場合，更新後のデータが表 T1 に存在するかどうか確認します。存在しないときは，先に表 T1 にデータを挿入後，表 T2 のデータを更新してください。

注※ 2

対策をする制約とは別の制約で，その表を被参照表とする参照表が存在する場合，修正順序に注意が必要です。例えば，次のような参照関係があるとします。




**● REF1 の制約違反の対策をする場合の注意**

T1 のデータを UPDATE 文で修正する場合，REF2 で ON UPDATE RESTRICT を指定しているときは，更新前のデータに対応する T3 のデータを削除後，T2 のデータを更新してください。

7. **違反があった制約を，強制的に検査保留状態に設定**

整合性チェックユティリティを制約単位で実行し，対策をした制約を強制的に検査保留状態に設定します。

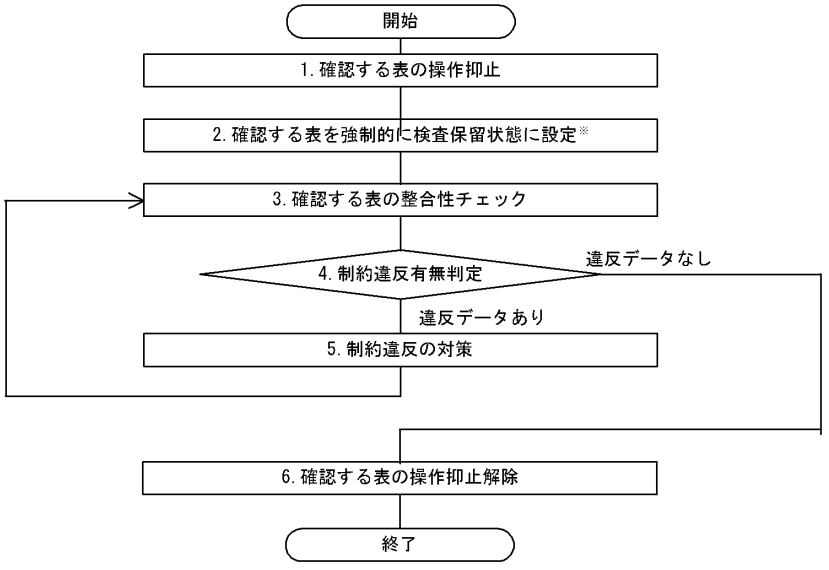
8. **確認する表の操作抑止解除**

運用を停止していた業務を再開します。手順 1. に戻り，整合性チェックを実行し，違反データがないことを確認します。

## (2) 検査保留状態を使用しない場合の整合性確認手順

pd_check_pending オペランドの指定値が NOUSE の場合の，整合性チェックユティリティを使用した整合性確認手順を次の図に示します。

**図 12-38 検査保留状態を使用しない場合の整合性確認手順**




注※ 制約名単位の整合性検査を実行する場合は不要です。

1. **確認する表の操作抑止**
   整合性が保証できない表を使用する業務の運用を停止します。
2. **確認する表を強制的に検査保留状態に設定**
   確認する表を強制的に検査保留状態に設定します。なお，手順 3. で制約単位の整合性チェックをする場合は，この操作は不要です。
3. **確認する表の整合性チェック**
   表単位，又は制約単位に整合性チェックをします。
4. **制約違反有無判定**
   手順 3. の整合性チェック結果で，制約違反データの有無を判定します。
5. **制約違反の対策**
   「12.19.5(1) 検査保留状態を使用する場合の整合性確認手順（参照制約）」の手順 6. を参照して，同様に制約違反データを修正してください。
6. **確認する表の操作抑止解除**
   運用を停止していた業務を再開します。

## 12.19.6 参照制約とトリガ

### (1) 参照制約動作のトリガ

参照制約動作に CASCADE を指定すると，HiRDB が内部的に参照表を更新するトリガを被参照表に対して生成します。HiRDB が内部的に生成するトリガは，次の場合に無効になるため，再作成する必要があります。ただし，HiRDB が生成したトリガだけを再作成することはできません。ALTER ROUTINE を使用して無効になったトリガすべてを再作成してください。

● 更新の場合
- 参照表の表定義を変更した場合

- 参照表にインデクスを定義した場合
- 参照表のインデクスを削除した場合
- 参照表に，トリガ契機が UPDATE のトリガを生成する場合
- 参照表のトリガ契機が UPDATE のトリガを削除する場合
- 参照表が参照する被参照表の主キー構成列の表定義を変更した場合

● 削除の場合

- 参照表の表定義を変更した場合
- 参照表にインデクスを定義した場合
- 参照表のインデクスを削除した場合
- 参照表に，トリガ契機が DELETE のトリガを生成する場合
- 参照表のトリガ契機が DELETE のトリガを削除する場合

また，HiRDB が内部的に生成するトリガは，参照表削除（DROP TABLE，又は DROP SCHEMA）時に削除されます。

### （2） 参照制約とユーザが定義したトリガの関係

トリガや参照制約が定義されている表に，更新系 SQL（INSERT 文，UPDATE 文，又は DELETE 文）を実行する場合の，トリガ，参照制約の整合性チェック，及び参照制約動作（HiRDB が参照制約定義時に内部的に生成するトリガ）の動作順序について説明します。これらの動作順序は，条件によって二つのパターンがあります。

パターン 1 の条件：

更新対象が被参照で参照制約動作の指定が RESTRICT だけの場合と，更新対象が参照表の場合

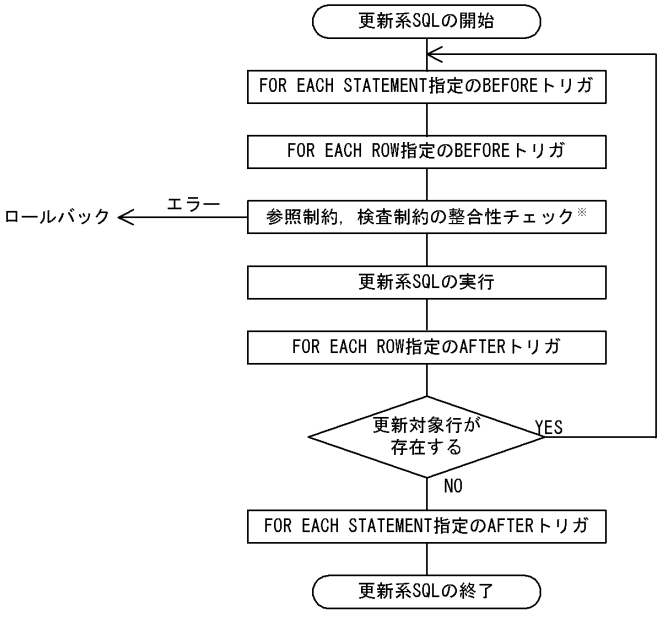
パターン 2 の条件：

更新対象が被参照で参照制約動作の指定に RESTRICT 以外がある場合

なお，更新対象の表が参照表であり，かつ被参照表でもある場合は，被参照表の条件が優先されます。

二つのパターンの場合の動作順序をそれぞれ次に示します。

**パターン**1

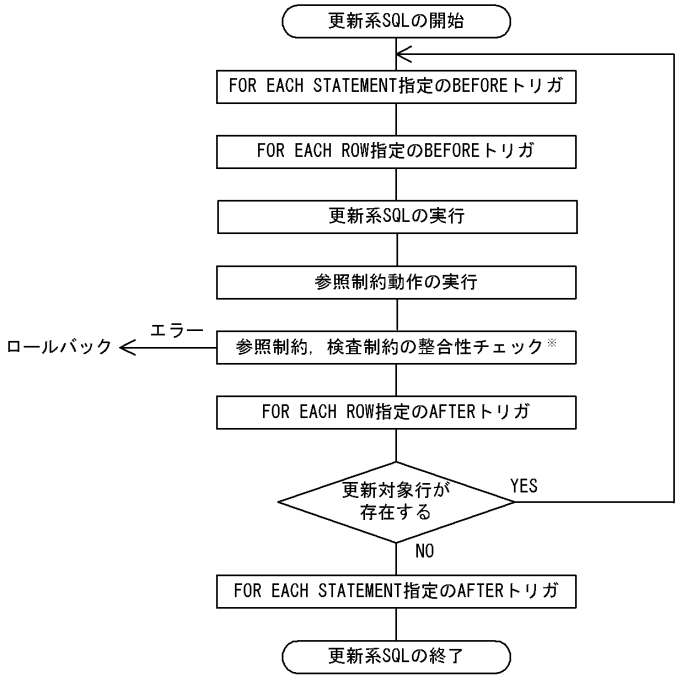



注※

参照制約の整合性はすべてこの時点でチェックされます。チェック内容を次に示します。

1. 更新対象が参照表の場合
   更新（INSERT，UPDATE）データが被参照表に含まれているかどうか
2. 更新対象が被参照表の場合
   更新（UPDATE，DELETE）データが参照表に含まれているかどうか
3. 更新対象が参照表で，かつ被参照表の場合
   上記 1，2 のチェック内容

**パターン** 2




注※

参照制約の整合性はすべてこの時点でチェックされます。チェック内容はパターン 1 と同じです。

## 12.19.7 関連製品との連携時の注意

関連製品との連携時の注意事項を次に示します。

● **HiRDB Datareplicator を使用する場合**

反映側の表に参照制約を定義しないでください。

● **分割格納条件を変更する場合**

被参照表の分割格納条件を変更する場合，及び既存データを削除するような RD エリアの統合又は分割をした場合，分割格納条件変更の完了後の整合性は保証されないため，データの整合性を確認する必要があります。データの整合性確認手順については，「12.19.5　表の整合性確認手順」を参照してください。

# 12.20　検査制約

## 12.20.1　検査制約とは

データベース中の表のデータは，値の範囲や条件など制限を持つ場合が多くあります。例えば，商品の情報をデータベースに格納する場合，商品価格として負の値はあり得ません。そのため，負の値はデータベースに存在してはいけない値であり，挿入又は更新時に値をチェックする必要があります。このように，データ挿入又は更新時に制約条件をチェックし，条件を満たさないデータの場合は操作を抑止することで表データの整合性を保つ制約が**検査制約**です。また，このマニュアルでは検査制約を定義した表を**検査制約表**といいます。

なお，ユティリティの実行などで検査制約表のデータの整合性が保証できなくなる場合があります。この場合，検査制約表は検査保留状態になります。検査保留状態については「12.20.3　検査保留状態」を，整合性を保証できなくなる操作については「12.20.4　データ操作と整合性」を参照してください。

**検査制約の効果**

検査制約を定義すると，データの挿入又は更新時のチェックを自動化できるので UAP を作成するときの負荷を軽減できます。ただし，検査制約表を更新する場合，データの整合性をチェックするため，チェックに掛かる処理時間が増加します。

## 12.20.2　検査制約の定義

検査制約は，定義系 SQL の CREATE TABLE で CHECK を指定し，表の値の制約条件を探索条件で指定します。また，検査保留状態を使用するには，pd_check_pending **オペランド**に USE を指定するか，又はオペランドの指定を省略します。

### (1) 検査制約を定義する表の制限事項

検査制約を定義する表の表定義，及び表定義変更時の制限事項を次に示します。

**(a) 表定義（CREATE TABLE）時**

- 検査制約は改竄防止表には定義できません。
- 検査制約は一つの表に 254 個まで定義できます。255 個以上は定義できません。例を次に示します。

```
CREATE TABLE T1(C001 INT CONSTRAINT CHECK_T1_C001 CHECK(C001>0),
                C002 INT CONSTRAINT CHECK_T1_C002 CHECK(C002>0),
                                  :
                C254 INT CONSTRAINT CHECK_T1_C254 CHECK(C254>0))
                C255 INT CONSTRAINT CHECK_T1_C255 CHECK(C255>0))
```

255個

この場合，検査制約数が 254 個より多いため，定義できません。表定義時にエラーとなります。

- 検査制約，及び各検査制約の探索条件中の AND，OR の個数の和は一つの表に 254 個まで定義できます（ただし，CASE 式中の探索条件，及びその探索条件中の AND，OR の個数は除きます）。例を次に示します。

```
                                                AND，ORの数200
CREATE TABLE T1(C001 INT CONSTRAINT CHECK_T1_C1 CHECK(C001=0 OR C001=1 OR ～C001=200),
                C002 INT CONSTRAINT CHECK_T1_C2 CHECK(C002=0 OR C002=1 OR ～C002=53))
                                                AND，ORの数53
```

この場合，検査制約数は 2 個ですが，制約名 CHECK_T1_C1 の探索条件中の AND の数が 200，制約

名 CHECK_T1_C2 の探索条件中の AND の数 53 で，検査制約数と各検査制約の探索条件中の AND，OR の和が 255（2 ＋ 200 ＋ 53）となり，254 個より多いため定義できません。表定義時にエラーとなります。
なお，表に定義されている検査制約数と各検査制約の探索条件中の AND，OR の和はディクショナリ表の SQL_TABLE 表の，N_CHECK_LIMIT 列に格納されています。

**(b) 表定義変更（ALTER TABLE）時**

- 検査制約表に対して，DROP 句及び RENAME 句を使用した表定義変更はできません。
- 検査制約表に対して，CHANGE 句を使用した次の変更はできません。
  - データ型やデータ長の変更
  - SPLIT の変更
  - 既定値の設定，解除
  - WITH DEFAULT の設定
- 検査制約表に対して，RENAME 句を使用した列名変更はできません。

### (2) 検査制約を定義する場合の注意事項

● SQL オブジェクト用バッファ長の容量見積もり
検査制約表に対して操作する場合，HiRDB が制約条件をチェックするトリガを生成します。そのため，SQL オブジェクト用バッファを指定するときに HiRDB が生成する制約条件の SQL オブジェクトについても考慮する必要があります。SQL オブジェクト用バッファ長（pd_sql_object_cache_size）の見積もり式については，マニュアル「HiRDB Version 8 システム定義」を参照してください。

● バックアップの取得
バックアップ取得時点の検査保留状態によって，バックアップの取得範囲が異なります。バックアップの取得時点と取得範囲については，マニュアル「HiRDB Version 8 システム運用ガイド」の「同時にバックアップを取得する必要がある RD エリア」を参照してください。

● データディクショナリ用 RD エリアの再編成
検査制約表の定義と削除を繰り返すと，データディクショナリ用 RD エリアの格納効率が低下します。このような場合，データベース状態解析ユティリティ（pddbst）でデータディクショナリ用 RD エリアの格納効率を確認し，必要に応じて再編成してください。

## 12.20.3 検査保留状態

ユティリティの実行などでデータの整合性を保証できなくなった場合，HiRDB は検査制約表に対するデータ操作を制限します。このように，整合性を保証できないためにデータ操作を制限された状態を**検査保留状態**といいます。検査制約表を検査保留状態にして，データ操作を制限するためには，pd_check_pending オペランドに USE を指定するか，又はオペランドの指定を省略する必要があります。検査保留状態の表は，整合性チェックユティリティ（pdconstck）を使用して検査保留状態を解除します。整合性チェックユティリティを使用して，強制的に検査保留状態に設定することもできます。

pd_check_pending オペランドに NOUSE を指定していると，データの整合性を保証できない場合でもデータ操作を制限しません。そのため，整合性が保証できなくなるユティリティを実行した場合は，整合性チェックユティリティで強制的に検査保留状態に設定してから，整合性を確認してください。

整合性が保証できなくなる操作については「12.20.4　データ操作と整合性」を，整合性の確認手順は「12.20.5　表の整合性確認手順」を参照してください。

### (1) 検査保留状態の管理

検査保留状態は**ディクショナリ表**と，表が格納された RD **エリアの表情報**で管理しています。ディクショナリ表では表単位，及び制約単位に検査保留状態を管理し，表情報では分割表の場合は RD エリア単位に，分割表ではない場合は表単位に検査保留状態を管理します。

検査保留状態の情報が格納されている場所と内容を次の表に示します。

**表 12-22 検査保留状態の情報が格納されている場所と内容（検査制約）**

| 格納場所 | | | 格納されている情報 |
|---|---|---|---|
| ディクショナリ表 | SQL_TABLES 表 | CHECK_PEND2 列 | 表単位の検査制約の検査保留状態 |
| | SQL_CHECKS 表 | CHECK_PEND2 列 | 制約単位の検査制約の検査保留状態 |
| RD エリアの表情報 | | 分割していない表の場合 | 表単位の参照制約又は検査制約の検査保留状態 |
| | | 分割表の場合 | RD エリア単位の参照制約又は検査制約の検査保留状態 |

### (2) 検査保留状態の表に対して制限される操作

参照制約の場合と同じです。「12.19.3(3) 検査保留状態の表に対して制限される操作」を参照してください。

### (3) 分割表を使用している場合

参照制約の場合と同じです。「12.19.3(5) 分割表を使用している場合」を参照してください。ただし，「参照表」を「検査制約表」に読み替えてください。

### (4) 検査保留状態を使用する場合の注意

- pd_check_pending オペランドの指定値を NOUSE から USE に変更した場合，整合性チェックユティリティを使用して，検査制約表の整合性を確認する必要があります。確認手順については，「12.20.5 表の整合性確認手順」を参照してください。
- pd_check_pending オペランドに USE を指定する場合，検査保留状態に設定される参照表及び RD エリアに対して排他が掛かるため，ユティリティ及び SQL 実行時の排他資源が検査保留状態を使用しない場合とは異なります。

## 12.20.4 データ操作と整合性

検査制約表に対する操作系 SQL による更新，追加，又は削除は，HiRDB が SQL 実行時にチェックし，整合性を保証します。ただし，次の表に示すユティリティによる操作をした場合，チェックをしないため，整合性を保証できなくなることがあります。pd_check_pending オペランドに USE を指定していて，これらの操作をした場合，検査制約表は検査保留状態になります。

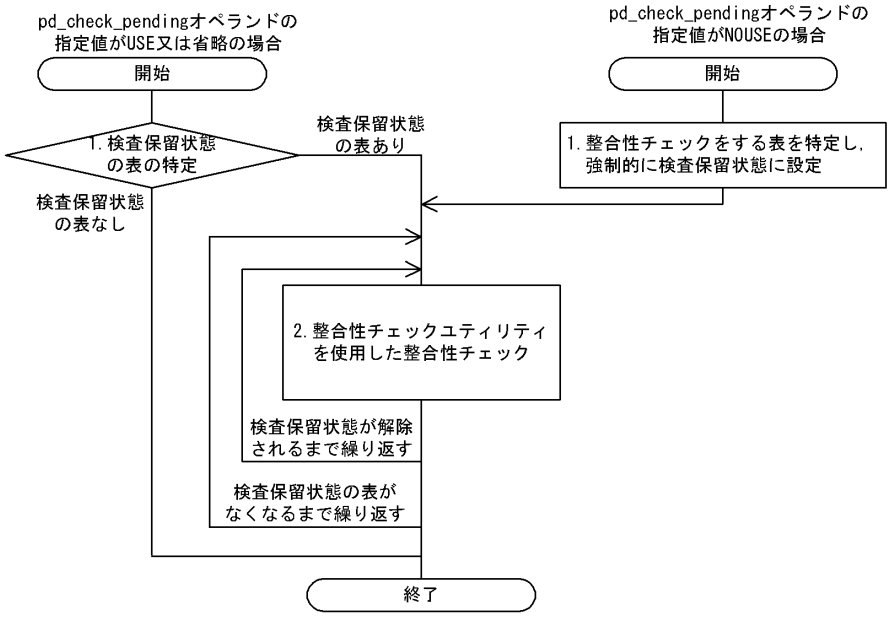
**表 12-23 整合性を保証できなくなる場合の，検査制約表に対する操作とデータ不整合の発生条件**

| 表又は RD エリアに対する操作 | | データ不整合の発生条件 |
|---|---|---|
| データベース作成ユティリティ（pdload） | データロード | 検査制約定義で指定した探索条件を満たさないデータをデータロードした場合 |
| データベース再編成ユティリティ（pdrorg） | リロード（-k reld 指定） | 検査制約定義で指定した探索条件を満たさないデータをリロードした場合 |

## 12.20.5 表の整合性確認手順

データの整合性確認手順の概要を次の図に示します。

**図 12-39 データの整合性確認手順の概要（検査制約）**




pd_check_pending **オペランドの指定値が** USE **又は省略の場合**

1. **検査保留状態の表の特定**
   ディクショナリ表の SQL_TABLES 表を検索して，検査保留状態の表名を検出します。

```
SELECT TABLE_SCHEMA, TABLE_NAME FROM MASTER.SQL_TABLES
    WHERE CHECK_PEND = 'C' OR CHECK_PEND2 = 'C'
```

   検索結果には，検査保留状態の表の所有者と検査保留状態の表名が返されます。検索結果が 0 行の場合，検査保留状態の表はありません。

2. **整合性チェックユティリティを使用した整合性チェック**
   整合性チェックユティリティで表単位の整合性チェックを実行し，制約違反データがあれば修正します。検査保留状態の表がなくなるまで整合性チェックを繰り返し，なくなれば終了です。整合性チェックユティリティを使用した整合性確認手順については，「12.20.5(1) 検査保留状態を使用する場合の整合性確認手順（検査制約）」を参照してください。

pd_check_pending **オペランドの指定値が** NOUSE **の場合**

1. **整合性チェックをする表を特定し，強制的に検査保留状態に設定**
   整合性チェックをする表を特定するために，整合性を保証できなくなる操作をした表に検査照制約が定義されているかどうかを確認します。これを確認する SQL の実行例を次に示します。

```
SELECT N_CHECK FROM MASTER.SQL_TABLES
    WHERE TABLE_SCHEMA = '対象表の所有者名' AND TABLE_NAME = '対象表の表名'
```

次の検索結果が返されます。

- 検査制約の定義数

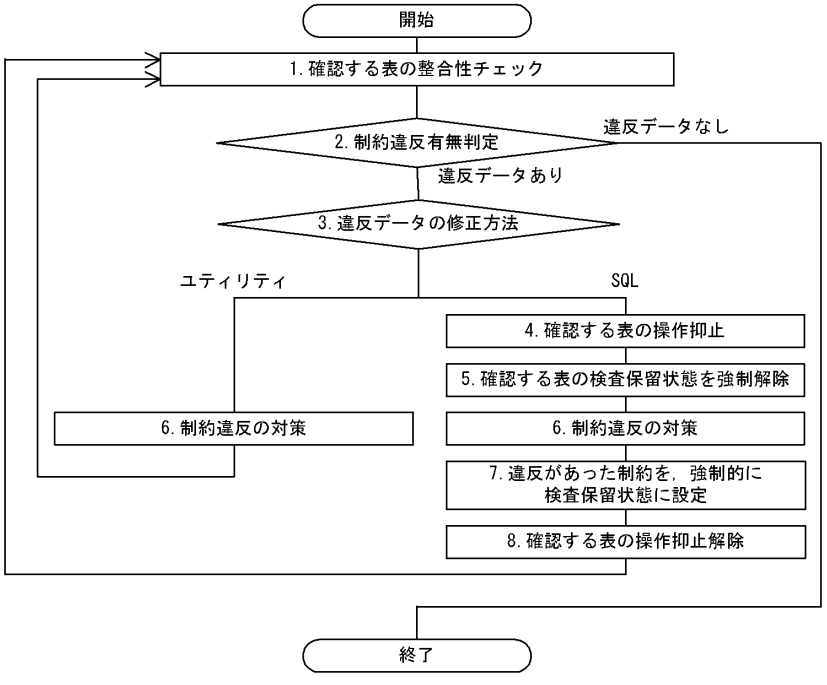
N_CHECK がナル値の場合，対象表に検査制約は定義されていません。
表を特定したら，整合性チェックユティリティを使用して，その表を強制的に検査保留状態に設定します（検査保留状態でない表は，整合性チェックユティリティでチェックできません）。

2. **整合性チェックユティリティを使用した整合性チェック**
pd_check_pending オペランドの指定値が USE 又は省略の場合の手順 2. と同じです。整合性チェックユティリティを使用した整合性確認手順は，参照制約の場合と同じのため，「12.19.5(2) 検査保留状態を使用しない場合の整合性確認手順」を参照してください。

## (1) 検査保留状態を使用する場合の整合性確認手順（検査制約）

pd_check_pending オペランドの指定値が USE 又は省略の場合の，整合性チェックユティリティを使用した整合性確認手順を次の図に示します。

**図 12-40 検査保留状態を使用する場合の整合性確認手順（検査制約）**




1. **確認する表の整合性チェック**
表単位，又は制約単位に整合性チェックをします。

2. **制約違反有無判定**
手順 1. の整合性チェック結果で，制約違反データの有無を判定します。

3. **違反データの修正方法**

違反データの修正をユティリティで行うか，SQL で行うかを選択します。ユティリティで行う場合，手順 6. へ進んでください。

4. **確認する表の操作抑止**
   整合性が保証できない表を使用する業務の運用を停止します。

5. **確認する表の検査保留状態を強制解除**
   制約違反の対策をするため，検査保留状態を強制解除します。

6. **制約違反の対策**

   **ユティリティで修正する場合**

   対策方法を次に示します。対策後，手順 1. に戻り，整合性チェックを実行し，違反データがないことを確認し，終了します。

| 条件 | 対策方法 |
|---|---|
| 検査制約の定義で指定した探索条件を修正する場合 | 手順を次に示します。<br>1. 表のデータをすべてアンロードします。<br>2. DROP TABLE で表の定義を削除します。<br>3. CREATE TABLE で表を再定義します。このとき，検査制約の定義に正しい探索条件を指定します。<br>4. 1. でアンロードしたデータをロードします。 |
| 表に制約違反データがある場合 | • データベース作成ユティリティ（pdload）の作成モードで正しいデータをロードします。<br>• データベース再編成ユティリティ（pdrorg）の UOC を使用した削除で不要なデータを削除します。 |

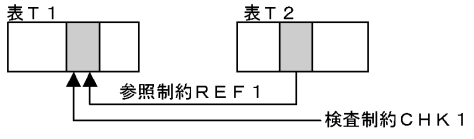
   **SQL で修正する場合**

   対策方法を次に示します。対策後，手順 7. に進みます。

| 条件 | 対策方法 |
|---|---|
| 検査制約の定義で指定した探索条件を修正する場合 | ユティリティで修正する場合と同じです。 |
| 表に制約違反データがある場合 | 制約違反データを DELETE 文で削除，又は，UPDATE 文で正しい値に更新します※。 |

   注※

   対策する表を被参照表とする参照表が存在する場合，修正順序に注意が必要です。例えば，次のような関係があるとします。




   **● CHK1 の制約違反の対策をする場合の注意**

   表 T1 のデータを DELETE 文で修正する場合，REF1 で ON DELETE RESTRICT を指定しているときは，対応する表 T2 のデータを先に削除後，表 T1 のデータを削除してください。また，UPDATE 文で修正する場合，REF1 で ON UPDATE RESTRICT を指定しているときは，更新前のデータに対応する表 T2 のデータを削除後，表 T1 のデータを更新してください。

7. **違反があった制約を強制的に検査保留状態に設定**
   整合性チェックユティリティを制約単位で実行し，対策をした制約を強制的に検査保留状態に設定します。

8. **確認する表の操作抑止解除**
   運用を停止していた業務を再開します。手順 1. に戻り，整合性チェックを実行し，違反データがないことを確認します。

### 12.20.6 関連製品との連携時の注意

関連製品との連携時の注意事項を次に示します。

● **HiRDB Datareplicator を使用する場合**
整合性があるデータが反映されるので，反映側の表に検査制約の定義は不要です。

# 13 インデクスの設計

この章では，B-tree 構造のインデクス及びプラグインインデクスを設計する上で検討する項目について説明します。


13.1 インデクスを設計するときの検討項目

13.2 インデクス

13.3 インデクスの横分割

13.4 プラグインインデクス

13.5 プラグインインデクスの横分割

# 13.1 インデクスを設計するときの検討項目

表に対する処理性能を向上させるには，インデクスを作成します。ただし，効果的に作成しないと，逆に性能を劣化させることもあります。このため，より効果的なインデクスの作成方法を検討する必要があります。また，インデクスのユーザ用 RD エリアへの格納の仕方によって，表に対する処理性能や操作性が異なります。これらの点を考慮してインデクスを設計する必要があります。

インデクスを設計するときの検討項目を次の表に示します。

**表 13-1 インデクスを設計するときの検討項目**

| 設計作業ごとの検討項目 | 長所 | 短所 | 記載箇所 |
|---|---|---|---|
| インデクスの作成 | 表の検索性能が向上します。 | 作成したインデクスの数が増えると，インデクスの更新処理時のオーバヘッドが増加します。 | 13.2 |
| インデクスの横分割 | 表格納 RD エリアとインデクス格納 RD エリアが 1 対 1 で管理できるため，ユティリティの操作性が向上します。 | 非分割キーインデクスを横分割すると，インデクスを使用した検索性能が悪くなることがあります。 | 13.3 |
| プラグインインデクスの作成 | 抽象データ型で定義された列にプラグインで提供されたインデクス型を指定してプラグインインデクスを作成すると，表の検索性能が向上します。 | 作成したインデクスの数が増えると，インデクスの更新処理時のオーバヘッドが増加します。 | 13.4 |
| プラグインインデクスの横分割 | インデクスの一括作成をするときに，ユーザ LOB 用 RD エリアごとに独立した運用ができます。 | 横分割すると RD エリアの数が増えます。RD エリア指定のデータベースのバックアップ及びデータベースの再編成時には，表とインデクスの対応が 1 対 1 であることに注意する必要があります。 | 13.5 |

# 13.2 インデクス

ここでは，B-tree 構造のインデクスの設計について説明します。

## 13.2.1 インデクスの作成

### (1) インデクスの効果

**性能の向上**

表を検索するときのキーとなる列にインデクスを作成しておくと，表の検索性能が向上します。

### (2) 適用基準

インデクスを作成するとよい列を次に示します。

- データを絞り込むための条件に使用する列
- 表の結合処理の条件として使用する列
- データのソート又はグループ分けの条件として使用する列
- 参照制約を定義した構成列（外部キー）

なお，次に示す列にインデクスを作成すると検索性能が低下するため，インデクスを作成しないようにします。

- 更新頻度の高い列
- データの重複度が高い列

### (3) 作成方法

表にインデクスを作成するには，定義系 SQL の CREATE INDEX を実行します。

### (4) 共通規則

1. 一つの表に最大 255 個のインデクスを定義できます。
2. ナル値を含む列又は行のない列に対してもインデクスを定義できます。
3. ビュー表にはインデクスを作成できません。
4. コストベースの最適化をする場合，最適化の精度を更に向上させるため，必要に応じて最適化情報収集ユティリティ（pdgetcst コマンド）で最適化情報を収集します。最適化情報収集ユティリティの実行要否については，マニュアル「HiRDB Version 8 コマンドリファレンス」の「最適化情報収集レベル」を参照してください。

### (5) インデクスを定義できないデータ型

次に示すデータ型の列にはインデクスを定義できません。

- 精度が 20 けた以上の DECIMAL
- BLOB
- BINARY
- BOOLEAN
- 抽象データ型

### (6) インデクスのキー長の上限

インデクスのキー長は，次に示す条件を満たす必要があります。この条件を満たさないとインデクスを定義できません。

```
インデクスのキー長（バイト）
    ≦ MIN｛(インデクス格納 RD エリアのページサイズ÷ 2）－ 1242，4036｝
```

インデクス格納 RD エリアのページサイズが 4096 バイトの場合は，キー長が最大 806 バイトのインデクスが定義できます。インデクスのキー長については，「表 16-5　インデクスのキー長一覧」を参照してください。

なお，複数列インデクスの場合は，複数列インデクスを構成する各列のキー長の合計がインデクスのキー長となります。

### (7) 注意事項

一つの表に同じインデクスを二つ以上作成できません。異なるインデクス名であっても同じインデクスと扱われる場合の例を次に示します。

**●単一列インデクスの場合**

```
CREATE INDEX インデクス1 ON 表1 (列1 ASC)
CREATE INDEX インデクス2 ON 表1 (列1 DESC)
```

この場合，インデクス 2 はインデクス 1 と同じインデクスとして扱われます。このため，先に定義したインデクス 1 が有効になります。

**●複数列インデクスの場合**

```
CREATE INDEX インデクス1 ON 表1 (列1 ASC,列2 ASC)
CREATE INDEX インデクス2 ON 表1 (列1 DESC,列2 DESC)
```

又は

```
CREATE INDEX インデクス1 ON 表1 (列1 ASC,列2 DESC)
CREATE INDEX インデクス2 ON 表1 (列1 DESC,列2 ASC)
```

この場合，インデクス 1 とインデクス 2 は同じインデクスとして扱われます。このため，先に定義したインデクス 1 が有効になります。なお，次に示す場合は，互いに異なるインデクスになります。

```
CREATE INDEX インデクス1 ON 表1 (列1 DESC,列2 DESC)
CREATE INDEX インデクス2 ON 表1 (列1 ASC,列2 DESC)
```

## 13.2.2 コストベースの最適化を考慮したインデクスの作成

HiRDB は，ある表にインデクスが複数作成されている場合，表の検索で指定された探索条件を基にして，最もアクセスコストの少ないインデクスを選択します。このようにインデクスを選択する処理を**コストベースの最適化**といいます。

HiRDB が見積もるアクセスコストを次に示します。

- 指定された探索条件によるヒット率
- SQL 処理に掛かる入出力処理の回数
- SQL 処理に掛かる CPU 負荷

HiRDB はコストベースの最適化をするため，表の検索性能が良くなります。特に，複雑な探索条件を指定した SQL 文を実行する場合でも，表の検索性能を低下させません。

## (1) コストベースの最適化を考慮したインデクスの作成基準

HiRDB はコストベースの最適化をするため，HiRDB が使用するインデクスの優先順位を意識しないで UAP を作成できます。ただし，UAP からアクセスする表に対してどのようにインデクスを作成しておくかについては，あらかじめ検討する必要があります。

コストベースの最適化を考慮するには，HiRDB が使用するインデクスの優先順位を考慮してインデクスを作成します。このとき，次に示す単一列インデクスと複数列インデクスの使い分け，複数のインデクスの使用及びインデクスの数が性能に与える影響も考慮してください。

HiRDB が使用するインデクスの優先順位を次の表に示します。

**表 13-2 HiRDB が使用するインデクスの優先順位**

| 優先順位 | HiRDB が優先して使用するインデクスの内容 | インデクスがある列（C1）に対する条件の指定例 |
|---|---|---|
| 1（必ず使用する※1） | インデクス型プラグイン専用関数の条件が「IS TRUE」であり，この関数の第一引数の列に指定したプラグインインデクス | `contains(C1,'…') IS TRUE` |
| | 構造化繰返し述語の探索条件に含まれるすべての列をインデクス構成列に含むインデクス | `ARRAY(C1,C2)[ANY]`<br>`  (C1='ABC' and C2=10)`<br>C1，C2 の複数列インデクスを定義しています。 |
| 2 | プラグイン提供関数の条件が「IS TRUE」であり，この関数の第一引数の列に指定したプラグインインデクス | `within(C1,'…') IS TRUE` |
| 3 | 「=」の制限条件の対象になる列にある，UNIQUE を指定したインデクス | `C1=100` |
| 4 | 「=」の制限条件の対象になる列にあるインデクス | `C1=100` |
| 5 | 「IS NULL」の制限条件の対象になる列にあるインデクス※2 | `C1 IS NULL` |
| 6 | LIKE 述語，又は SIMILAR 述語のパターン文字列に定数で「%」による前方一致比較を指定した列にあるインデクス | `C1 LIKE 'ABC%'`<br>`C1 SIMILAR TO 'ABC%'` |
| 7 | LIKE 述語，又は SIMILAR 述語のパターン文字列に定数で上記以外の前方一致比較を指定した列にあるインデクス | `C1 LIKE 'ABC_'`<br>`C1 SIMILAR TO 'ABC_'` |
| 8 | IN 述語の制限条件の対象になる列にあるインデクス | `C1 IN(10,20,30)` |
| 9 | BETWEEN 述語の制限条件の対象になる列にあるインデクス | `C1 BETWEEN 20 AND 40` |
| | 範囲条件を指定した列にあるインデクス | `20<=C1 AND C1<=40` |
| 10 | 外への参照がない副問合せを使用した IN 述語の制限条件の対象になる列にある単一列インデクス | `C1 IN(SELECT C1 FROM T2)` |
| | 外への参照がない副問合せを使用した限定述語「=ANY」又は「=SOME」の制限条件の対象になる列にある単一列インデクス | `C1=ANY(SELECT C1 FROM T2)`<br>`C1=SOME(SELECT C1 FROM T2)` |
| 11 | >，>=，< 及び <= の制限条件の対象になる列にあるインデクス | `C1>50`<br>`C1<=200` |
| 12※3 | スカラ演算（システム定義スカラ関数，IS_USER_CONTAINED_IN_HDS_GROUP は除く）を指定した列にあるインデクス※2 | `length(C1)=10` |

| 優先順位 | HiRDB が優先して使用するインデクスの内容 | インデクスがある列（C1）に対する条件の指定例 |
|---|---|---|
| 13 | NOT BETWEEN 述語の制限条件の対象になる列にあるインデクス | `C1 NOT BETWEEN 10 AND 30` |
| 14 | XLIKE 述語及び上記以外の LIKE 述語，又は SIMILAR 述語の制限条件の対象になる列にあるインデクス | `C1 XLIKE '%ABC%'`<br>`C1 LIKE '%ABC%'`<br>`C1 SIMILAR TO '%ABC%'` |
| 15 | 集合関数（MIN 又は MAX）の引数に指定した列にあるインデクス※4 | `MIN(C1)`<br>`MAX(C1)` |
| 16 | 結合条件列及びグループ分け又はソートの対象になる列にあるインデクス | `ORDER BY C1` |
| － | 否定（NOT BETWEEN を除く）の制限条件の対象になる列にあるインデクス | `C1 NOT LIKE '%ABC%'`<br>`C1 IS NOT NULL` |
| | 上記以外の限定述語「ANY」又は「SOME」の制限条件の対象になる列にあるインデクス | `C1>=ANY(SELECT C1 FROM T2)`<br>`C1>SOME(SELECT C1 FROM T2)` |
| | 限定述語「ALL」の制限条件の対象になる列にあるインデクス | `C1>ALL(SELECT C1 FROM T2)` |
| | プラグイン提供関数の条件が「IS FALSE」又は「IS UNKNOWN」であり，この関数の第一引数の列に指定したプラグインインデクス | `within(C1,'…') IS FALSE` |

（凡例）－：使用しないインデクスを示します。

注

1. 関数呼出し「contains」は，HiRDB Text Search Plug-in が提供するプラグインの関数です。
2. 関数呼出し「within」は，HiRDB Spatial Search Plug-in が提供するプラグインの関数です。
3. 外への参照がある副問合せを含む制限条件の対象になる列にあるインデクスは使用できません。
4. OR 演算子で両側の条件式にインデクスを利用できる場合は，優先度がその条件式中の述語によって変わります。
5. 制限条件とは，結合条件以外の探索条件のことです。
6. 定義したインデクスが有効に利用できないと HiRDB が判断した場合は，該当するインデクスが使用されない場合があります。

注※ 1

このインデクスを定義していないと，実行できないでエラーとなります。必ず定義してください。

注※ 2

次の列に対して，ナル値を除外キーとするインデクスは使用しません。

- 「IS NULL」の制限条件を指定した列
- 制限条件中の「VALUE」及び「CASE 式」に指定した列
- 制限条件中の「BIT_AND_TEST」に対して「IS UNKNOWN」，「IS NOT TRUE」，又は「IS NOT FALSE」を指定し，この「BIT_AND_TEST」に指定した列

ただし，上記以外の制限条件を指定しているインデクスは使用します。その場合のナル値を除外キーとするインデクスの使用有無を表 13-3 に示します。

注※ 3

SQL 最適化オプションに「スカラ演算を含むキー条件の適用」を指定した場合だけ，インデクスを優先して使用します。SQL 最適化オプションの詳細は，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。また，述語の種類によっては優先順位が下がることもあります。否定が含

まれない場合の優先順位は 12 ～ 14 の範囲になります。否定が含まれる場合の優先順位は 12 ～－の範囲になります。

注※ 4

表の指定が一つで，GROUP BY を指定していない SQL 文の場合は，集合関数（MIN 又は MAX）だけを指定して，次に示すどれかの条件を満たすときに引数に指定した列のインデクスを使用します。

- 集合関数の引数に指定した列が，単一列インデクスの構成列の場合
- 集合関数の引数に指定した列が，除外キーを持たない複数列インデクスの第 n 構成列の場合で，第 1 構成列から第（n － 1）構成列までに「=」又は「IS NULL」を指定したとき
- 集合関数の引数に指定した列が，除外キーを持つ複数列インデクスの第 n 構成列の場合で，第 1 構成列から第（n － 1）構成列までに「=」を指定したとき

**表 13-3　ナル値を除外キーとするインデクスの使用有無**

| 構成列に指定する制限条件 | | 使用有無 |
|---|---|---|
| IS NULL，VALUE，CASE 式，BIT_AND_TEST | IS NULL，VALUE，CASE 式，BIT_AND_TEST 以外※1 | |
| 指定あり | 指定あり | 使用します |
| 指定あり | 指定なし | 使用しません |
| 指定なし | 指定あり | 使用します※2 |
| 指定なし | 指定なし | 使用しません※3 |

注※ 1

「表 13-2　HiRDB が使用するインデクスの優先順位」に示す優先順位の 3 ～ 14 の制限条件の場合です。

注※ 2

インデクスが有効に利用できない場合などは，HiRDB が判断してインデクスを使用しないことがあります。

注※ 3

次に示すすべての条件を満たす検索では使用します。

- 選択式が，インデクス構成列を引数とする集合関数だけ
- FROM 句には，一つの表だけ指定している
- WHERE 句の指定がない

「表 13-2　HiRDB が使用するインデクスの優先順位」に示すインデクスの優先順位でインデクスを作成しておくと，この優先順位で SQL 文中に指定した探索条件での絞り込みの効果が期待できます。なお，優先順位の高いインデクスを作成していても，コストベースの最適化によって HiRDB がコストが高いと判断すると，このインデクスが使用されないことがあります。

## 13.2.3　単一列インデクスと複数列インデクスの使い分け

インデクスには，単一列インデクスと複数列インデクスがあります。**単一列インデクス**とは，表の一つの列で作成した一つのインデクスのことです。**複数列インデクス**とは，表の複数の列で作成した一つのインデクスのことです。

### (1) 単一列インデクスを作成する場合

一つの列をキーにして検索する場合に単一列インデクスを作成します。

## (2) 複数列インデクスを作成する場合

次に示す場合に，複数列インデクスを作成します。

### (a) 検索するデータを複数の条件を満たすデータだけに絞り込む場合

複数の列をキーとして，AND 演算子を使用した複合条件検索をする場合など，複数の条件を満たすデータだけに絞り込んで検索する場合は，複数列インデクスを作成します。

例えば，次に示す表の C1，C2 及び C3 の三つの列をキー項目として複合条件検索をするとします。

```
SELECT 検索する列 FROM 検索対象の表 WHERE C1=10 AND C2=20 AND C3=30
```

この場合は，C1，C2 及び C3 のそれぞれの列に単一列インデクスを作成するのではなく，C1，C2 及び C3 の三つの列で構成する複数列インデクスを作成します。これによって，インデクスに対するアクセスと行に対するアクセスのオーバヘッドを削減できます。

なお，複合条件検索をする場合，複数列インデクスの第 1 構成列は，必ず「=」条件を指定する列にします。また，「=」条件を指定する確率が高い列ほど先に指定します。これによって，インデクス内の検索範囲を小さくできるため，インデクス内の検索時間を短縮できます。しかし，複数列インデクスの第 1 構成列に「=」条件を指定しない場合は，十分な効果が得られないときがあります。このような場合は，単一列インデクスを使用した方が効果があるときがあります。

### (b) 探索条件でデータを絞り込んだ後，グループ分けやソートなどをする場合

探索条件として指定する列，グループ分け又はソートをする列の順で構成する複数列インデクスを作成します。

例えば，次に示す表の C1 と C2 の列をキーとして複合条件検索をして，検索結果を C3 の列で降順，C4 の列で昇順にソートするとします。

```
SELECT 検索する列 FROM 検索対象の表 WHERE C1=10 AND C2=20
  ORDER BY C3 DESC,C4 ASC
```

この場合も，C1 と C2 のそれぞれの列に単一列インデクスを作成するのではなく，C1，C2，C3 及び C4 で構成する複数列インデクスを作成します。このとき，C3 の列の並びが降順，C4 の列の並びが昇順になるようにします。これによって，インデクスに対するアクセスと行に対するアクセスのオーバヘッドを削減できます。

### (c) 一つの表に作成した複数の複数列インデクスが，それぞれある列で重複している場合

例えば，一つの表に，C1 と C2，C1 と C3 の列でそれぞれ構成する複数列インデクスを作成した場合，データを更新するとき，重複している列（C1 の列）を更新するためのオーバヘッドが大きくなります。このオーバヘッドを削減するには，C1，C2 及び C3 の列で構成する一つの複数列インデクスを作成します。

なお，C1 と C3 の列に探索条件を指定して検索した場合は，検索性能が低下することがあります。

### (d) 単一列インデクスと複数列インデクスの優先順位

表に単一列インデクスと複数列インデクスを作成している場合に，HiRDB が使用するそれぞれのインデクスの優先順位の例を次の表に示します。なお，次の表は，表を検索するときに C1=10 AND C2=20 を指定した場合の例です。

表 13-4 単一列インデクスと複数列インデクスの優先順位

| インデクスを構成する列 | | | 優先順位 |
|---|---|---|---|
| 第 1 構成列 | 第 2 構成列 | 第 3 構成列 | |
| C1 | C2 | なし | 1 |
| C1 | C3 | C2 | 2 |
| C1 | なし | なし | 3 |
| C1 | C3 | なし | 4 |
| C3 | C2 | なし | 5 |

## 13.2.4 複数のインデクスの使用

表には複数のインデクスを作成できます。一つのインデクス（単一列インデクス又は複数列インデクス）を使用するよりも複数のインデクスを使用した方が，より行を絞り込める場合に有効です。

## 13.2.5 除外キー値を設定したインデクスの使用

インデクスを定義した列のデータは，すべてインデクスのキー値としてインデクス中に取り込まれます。ところが，インデクス中にはナル値のような余分なキー値もあります。このように，すべての構成列の値がナル値から成るキー値の重複が多いインデクスに対してナル値を**除外キー値**として指定できます。

### (1) インデクスに除外キー値を設定した場合の効果

インデクスに除外キー値を設定することで，次に示す効果が期待できます。

1. インデクスには，ナル値のキーを作成しないため，インデクスの容量を削減できます。
2. 行の挿入，削除及び更新時のインデクスメンテナンスのオーバヘッド（CPU 時間，入出力回数，排他制御要求回数及びデッドロック発生頻度）とログ量を削減できます。
3. ナル値を除外キー値とするインデクスの構成列に対する探索条件が，IS NULL，VALUE，又は CASE 式を指定した検索ではインデクスを使用しません。これによって，次に示す場合に検索性能が向上します。
   - ナル値の重複が多い状態でインデクスを使用し，データページをランダムにアクセスしたため，同じページに入出力処理が発生していた場合

### (2) 設定方法

除外キー値を設定するには，定義系 SQL の CREATE INDEX に EXCEPT VALUES オプションを指定します。

### (3) 注意

- 除外キー値を指定できるのはナル値だけで構成されたキー値です。
- 非ナル値制約の列を含むインデクスには除外キー値を指定できません。
- クラスタキー指定を指定したインデクスには除外キー値を指定できません。
- インデクス順にアンロードをする場合，除外キー値を持つインデクスは指定できません。

## 13.2.6 インデクスの数が性能に与える影響

行を追加又は削除するとき，該当する表に作成しているすべてのインデクスが更新されます。このため，作成するインデクスの数が増えると，インデクスの更新処理のオーバヘッドが増加します。そのため，次に示す点を考慮してインデクスを作成します。

- 更新の多い列にはインデクスを定義しないようにします。
- 複数列インデクスを作成して，インデクスの数を減らします。
- HiRDB/ パラレルサーバの場合に，主に全件検索をするときは，並列処理の効果を上げるため，最低限の数だけインデクスを作成するようにします。

# 13.3 インデクスの横分割

表を横分割した場合，横分割した表に対応させて，インデクスも複数のユーザ用 RD エリアにわたって横分割できます。

## (1) 分割キーインデクスと非分割キーインデクス

横分割インデクスを設計する前に，**分割キーインデクス**と**非分割キーインデクス**について理解する必要があります。

インデクスがある一定の条件を満たすと，そのインデクスは分割キーインデクスになり，条件を満たさないインデクスは非分割キーインデクスになります。この条件は，表が単一列分割か複数列分割かによって異なります。

注

表の分割条件に一つの列だけを使用している場合を**単一列分割**といい，表の分割条件に複数の列を使用している場合を**複数列分割**といいます。

### (a) 単一列分割の場合

次に示すどちらかの条件を満たす場合，そのインデクスは分割キーインデクスになります。

**〈条件〉**

- 表を横分割するときに格納条件を指定した列（分割キー）に定義した単一列インデクス
- 表を横分割するときに格納条件を指定した列（分割キー）を第 1 構成列とした複数列インデクス

次に示す ZAIKO 表を例にして，インデクスが分割キーインデクスになる場合を次の図に示します。

**図 13-1 分割キーインデクスになる場合（単一列分割の場合）**

ZAIKO

| SCODE | SNAME | COL | TANKA | ZSURYO |
|---|---|---|---|---|
| 101L | ブラウス | 青 | 3500 | 62 |
| 101M | ブラウス | 白 | 3500 | 85 |
| 201M | ポロシャツ | 白 | 3640 | 29 |

分割条件に指定した列（分割キー）

〔説明〕

```
CREATE INDEX A12 ON ZAIKO (SCODE ASC)                1
CREATE INDEX A12 ON ZAIKO (SCODE ASC,TANKA DESC)     2
CREATE INDEX A12 ON ZAIKO (TANKA DESC,SCODE ASC)     3
```

1. 分割キーである SCODE 列をインデクスとした場合，そのインデクスは**分割キーインデクス**になります。そのほかの列をインデクスとした場合，そのインデクスは**非分割キーインデクス**になります。
2. 分割キーである SCODE 列を複数列インデクスの第 1 構成列にすると，その複数列インデクスは**分割キーインデクス**になります。
3. 分割キーである SCODE 列を第 1 構成列以外に指定すると，その複数列インデクスは**非分割キーインデクス**になります。

### (b) 複数列分割の場合

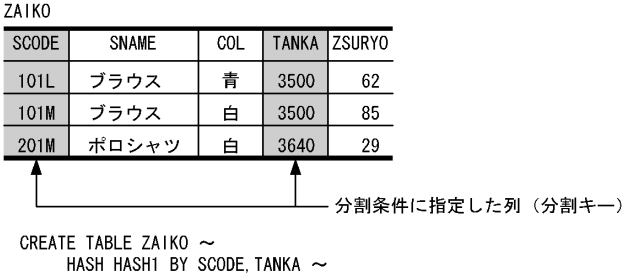
次に示す条件を満たす場合，そのインデクスは分割キーインデクスになります。

**〈条件〉**

- 分割キーを先頭とし，分割に指定した列を先頭から同順にすべて含んで，複数の列に作成したインデクス

次に示すZAIKO表を例にして，インデクスが分割キーインデクスになる場合を次の図に示します。

**図 13-2　分割キーインデクスになる場合（複数列分割の場合）**




〔説明〕

```
CREATE INDEX A12 ON ZAIKO (SCODE ASC,TANKA DESC)              1
CREATE INDEX A12 ON ZAIKO (SCODE ASC,TANKA DESC,ZSURYO ASC)   2
CREATE INDEX A12 ON ZAIKO (TANKA DESC,SCODE ASC)              3
CREATE INDEX A12 ON ZAIKO (SCODE ASC,ZSURYO DESC,TANKA ASC)   4
```

1. すべての分割キー（SCODE及びTANKA列）を指定し，かつ分割キーの指定順序が表定義時と同じため，この複数列インデクスは**分割キーインデクス**になります。
2. すべての分割キー（SCODE及びTANKA列）を指定し，かつ分割キーの指定順序が表定義時と同じため，この複数列インデクスは**分割キーインデクス**になります。
3. すべての分割キー（SCODE及びTANKA列）を指定しているが，分割キーの指定順序が表定義時と異なるため，この複数列インデクスは**非分割キーインデクス**になります。
4. すべての分割キー（SCODE及びTANKA列）を指定しているが，分割キーの指定順序が表定義時と異なるため，この複数列インデクスは**非分割キーインデクス**になります。

## (2) インデクスの分割指針

インデクスが分割キーインデクスか非分割キーインデクスかによって，次の表に示すとおりインデクスの分割指針が異なります。

**表 13-5　インデクスの分割指針**

| インデクスの種類 | HiRDB/シングルサーバの場合 | HiRDB/パラレルサーバの場合 | |
|---|---|---|---|
| | | 表をサーバ内横分割する場合 | 表をサーバ間横分割する場合 |
| インデクスが分割キーインデクスの場合 | 横分割表に対応させてインデクスも横分割します。 | 横分割表に対応させてインデクスも横分割します。 | 横分割表に対応させてインデクスも横分割します。 |
| インデクスが非分割キーインデクスの場合 | インデクスを横分割しないことをお勧めします。インデクスを横分割すると，インデクスを使用した検索性能が悪くなることがあります※。 | インデクスを横分割しないことをお勧めします。インデクスを横分割すると，インデクスを使用した検索性能が悪くなることがあります※。 | |

注※

非分割キーインデクスは横分割しないことをお勧めします。インデクスを横分割すると，インデクスを使用した検索性能が悪くなることがあります。具体的には，次に示すアクセスパスを使用した検索ができなくなるため，インデクスを使用した検索性能が悪くなることがあります。

- KEY SCAN MERGE JOIN
- LIST SCAN MERGE JOIN
- L-KEY R-LIST MERGE JOIN
- L-KEY R-SORT MERGE JOIN
- L-LIST R-KEY MERGE JOIN
- L-LIST R-SORT MERGE JOIN
- L-SORT R-KEY MERGE JOIN
- L-SORT R-LIST MERGE JOIN

これらのアクセスパスについては，マニュアル「HiRDB Version 8 コマンドリファレンス」のアクセスパス表示ユティリティ（pdvwopt コマンド）を参照してください。
ただし，表のデータが非常に多い場合は，インデクスの横分割を検討してください。インデクスを横分割すると，表格納 RD エリアとインデクス格納 RD エリアが 1 対 1 で管理できるため，ユティリティの操作性が向上します。例えば，インデクスを横分割しない場合に RD エリア単位のデータロード，又は RD エリア単位の再編成をしたときは，データロード又は再編成後にインデクスを一括作成する必要があります。インデクスを横分割すれば，RD エリア単位のデータロード，又は RD エリア単位の再編成後にインデクスを一括作成する必要はありません。
なお，マトリクス分割表にインデクスを定義する場合，非分割キーインデクスであっても分割キーと同様に横分割する必要があります。

### (3) 設計上の考慮点

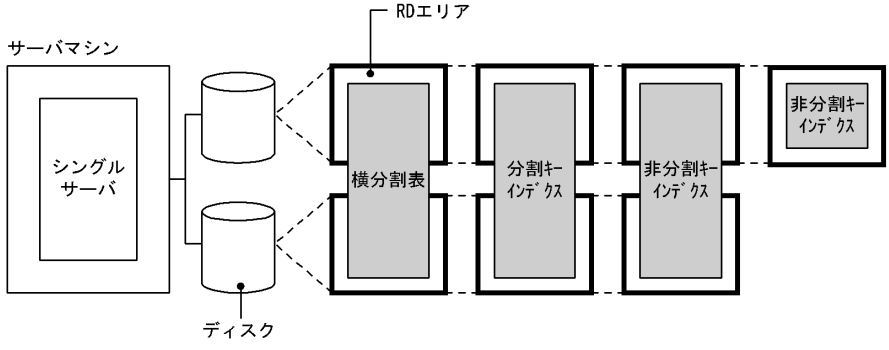
- 横分割表を格納するユーザ用 RD エリアと，横分割表のそれぞれに対応するインデクスを格納するユーザ用 RD エリアを分けます。これによって，それぞれのユーザ用 RD エリアの使用効率が良くなります。
- 表の中に一意にしたいキーがあるときは，このキーに UNIQUE を指定した分割キーインデクスを定義します。又は，分割キーに対してクラスタキーを指定します。また，非分割キーインデクスの場合でも，次のどちらかであれば UNIQUE を指定できます。
  - 非分割インデクス
  - 分割キーを任意の構成列に含む分割インデクス

  ただし，フレキシブルハッシュ分割した表では，インデクスに UNIQUE を指定できません。詳細については，マニュアル「HiRDB Version8 SQL リファレンス」の「CREATE INDEX」の「表を横分割する場合の UNIQUE 指定可否」を参照してください。

### (4) インデクスの横分割の例（HiRDB/ シングルサーバの場合）

インデクスの横分割の例（HiRDB/ シングルサーバの場合）を次の図に示します。

**図 13-3 インデクスの横分割の例（HiRDB/ シングルサーバの場合）**




〔説明〕

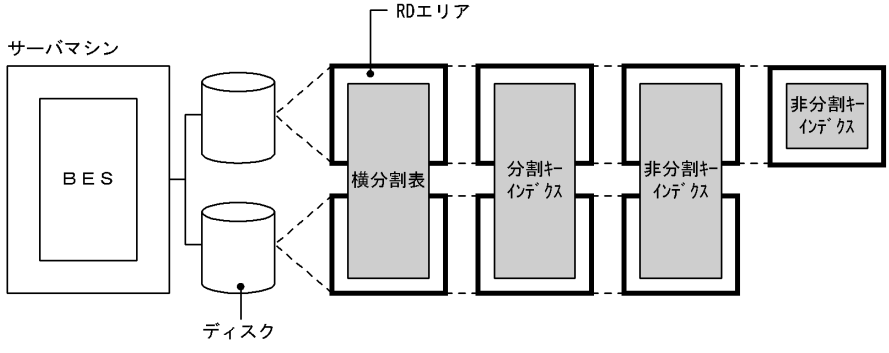
- ディスクのアクセス競合を避けるために，分割した表及びインデクスを格納する RD エリアを異なるディスク上に配置してください。
- 分割キーインデクスは横分割してください。
- 性能を重視する場合は，非分割キーインデクスを横分割しないでください。
- 操作性を重視する場合は，非分割キーインデクスを横分割してください。

## (5) インデクスの横分割の例（HiRDB/ パラレルサーバの場合）

### (a) 表をサーバ内横分割する場合

インデクスの横分割の例（サーバ内横分割の場合）を次の図に示します。

**図 13-4 インデクスの横分割の例（サーバ内横分割の場合）**




（凡例）ＢＥＳ：バックエンドサーバ

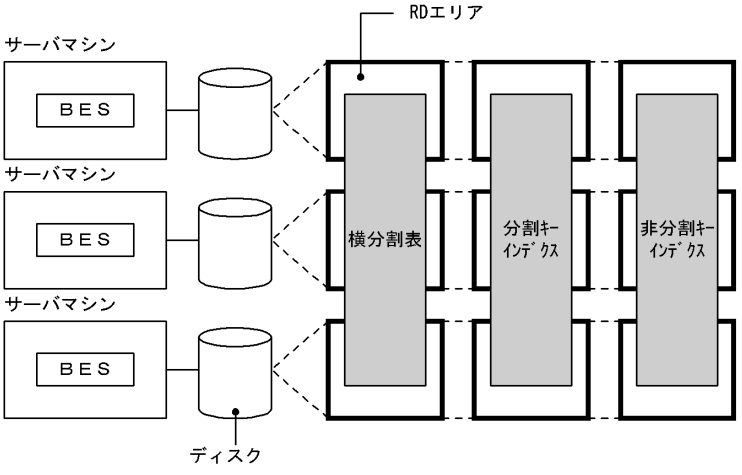
〔説明〕

- ディスクのアクセス競合を避けるために，分割した表及びインデクスを格納する RD エリアを異なるディスク上に配置してください。
- 分割キーインデクスは横分割してください。
- 性能を重視する場合は，非分割キーインデクスを横分割しないでください。
- 操作性を重視する場合は，非分割キーインデクスを横分割してください。

### (b) 表をサーバ間横分割する場合

インデクスの横分割の例（サーバ間横分割の場合）を次の図に示します。

**図 13-5 インデクスの横分割の例（サーバ間横分割の場合）**




（凡例）ＢＥＳ：バックエンドサーバ

〔説明〕

- ディスクのアクセス競合を避けるために，分割した表及びインデクスを格納する RD エリアを異なるディスク上に配置してください。
- 分割キーインデクス及び非分割キーインデクスを横分割してください。

# 13.4 プラグインインデクス

ここでは，プラグインインデクスについて説明します。

## (1) プラグインインデクスの効果

**性能の向上**

プラグインを使用している場合に，プラグインインデクスを作成すると，表の検索性能を向上できます。プラグインで提供されるインデクス型を使用すると，複雑な検索を高速にできます。

## (2) 作成方法

表にプラグインインデクスを作成するには，定義系 SQL の CREATE INDEX を実行します。

## (3) 注意

プラグインによっては，あらかじめ指定されたプラグインによるプラグインインデクスの定義を前提としていることがあります。その場合，プラグインインデクスを定義しないで，プラグインインデクスを利用する関数を指定すると，実行時にエラーが返されることがあります。

## (4) プラグインインデクスの一括作成

プラグインインデクスは，データベース作成ユティリティ（pdload）による一括作成ができます。プラグインインデクスの一括作成については，「6.4　プラグインが提供する抽象データ型を定義した表の作成」を参照してください。

# 13.5　プラグインインデクスの横分割

表を横分割した場合，横分割した表に対応させて，プラグインインデクスも複数のユーザ LOB 用 RD エリアにわたって横分割する必要があります。

## (1) プラグインインデクスの横分割の効果

**操作性の向上**

プラグインインデクスの一括作成をするときに，ユーザ LOB 用 RD エリアごとに独立した運用ができます。

## (2) 定義方法

プラグインインデクスの横分割の定義方法については，「6.4　プラグインが提供する抽象データ型を定義した表の作成」を参照してください。

## (3) プラグインインデクスの横分割の形態

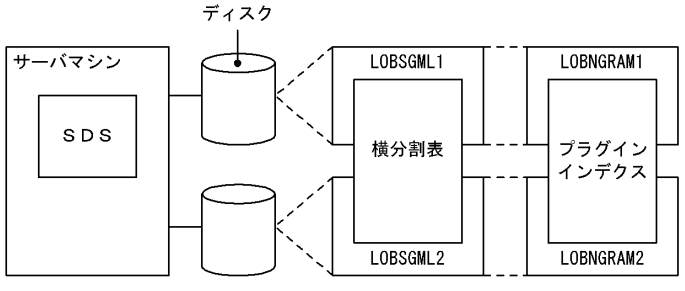
HiRDB/ シングルサーバと HiRDB/ パラレルサーバの，それぞれの場合でのプラグインインデクスの横分割の形態を次に示します。

### (a) HiRDB/ シングルサーバの場合

HiRDB/ シングルサーバの場合は，横分割した表に対応させて，複数のディスク上のユーザ LOB 用 RD エリアにわたってプラグインインデクスを横分割できます。

プラグインインデクスの横分割の形態を図 13-6 に示します。図 13-6 の形態でのプラグインインデクスの横分割の例を図 13-7 に示します。

**図 13-6　プラグインインデクスの横分割の形態（HiRDB/ シングルサーバの場合）**




（凡例）
ＳＤＳ：シングルサーバ
LOBSGML1～LOBSGML2及び
LOBNGRAM1～LOBNGRAM2：ユーザＬＯＢ用ＲＤエリア

注　表とプラグインインデクスの対応は１対１になります。

**図 13-7 プラグインインデクスの横分割（キーレンジ分割）の例（HiRDB/ シングルサーバの場合）**

● 薬品管理表をユーザＬＯＢ用ＲＤエリアLOBSGML1とLOBSGML2に横分割します。

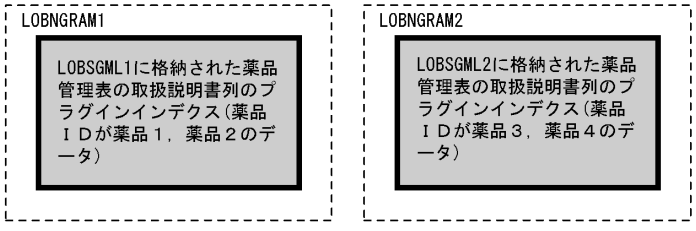
LOBSGML1 薬品管理表

| 薬品ＩＤ | 取扱説明書 |
|---|---|
| 薬品１ | <添付文書データ><効能>下痢，食あたり，水あたり，・・・</効能><br><用法・用量>大人(20才以上)1回10錠，・・・食後に服用する。</用法・用量><br>：<br><使用上の注意>小児の手の届かない･･･。冷蔵庫に保管･･･。</使用上の注意></添付文書データ> |
| 薬品２ | <添付文書データ><効能>頭痛，歯痛，神経痛，腰痛，・・・</効能><br><用法・用量>大人(20才以上)1回5包・・・服用間隔は24時間以上おいてください。</用法・用量><br>：<br><使用上の注意>小児の手の届かない･･･。頭痛以外の場合は･･･。</使用上の注意></添付文書データ> |
| ： | ： |

LOBSGML2 薬品管理表

| 薬品ＩＤ | 取扱説明書 |
|---|---|
| 薬品３ | <添付文書データ><効能>かぜ，かぜと思われる症状，・・・</効能><br><用法・用量>1日5回・・・食後なるべく5分以内に・・・</用法・用量><br>：<br><使用上の注意>日の当たらない場所に・・・。服用後は車等の・・・</使用上の注意></添付文書データ> |
| 薬品４ | <添付文書データ><効能>目の疲れ，肩こり，・・・</効能><br><用法・用量>成人(20才以上)1日10錠を・・・服用してください。</用法・用量><br>：<br><使用上の注意>小児の手の届かない･･･。</使用上の注意></添付文書データ> |
| ： | ： |

● 横分割した薬品管理表に対応してプラグインインデクスを横分割します。




〔説明〕

取扱説明書列にプラグインインデクスが設定してあるとします。

薬品管理表を薬品 ID を条件として，ユーザ LOB 用 RD エリア LOBSGML1，LOBSGML2 にわたって横分割しています。これに対応して，プラグインインデクスを LOBNGRAM1，LOBNGRAM2 に格納しています。

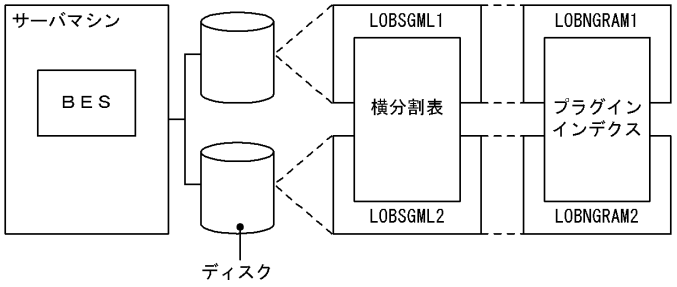
### (b) HiRDB/ パラレルサーバの場合

HiRDB/ パラレルサーバの場合は，横分割した表に対応して，複数のサーバマシン又はバックエンドサーバに配置されたユーザ LOB 用 RD エリアにわたってプラグインインデクスを横分割できます。

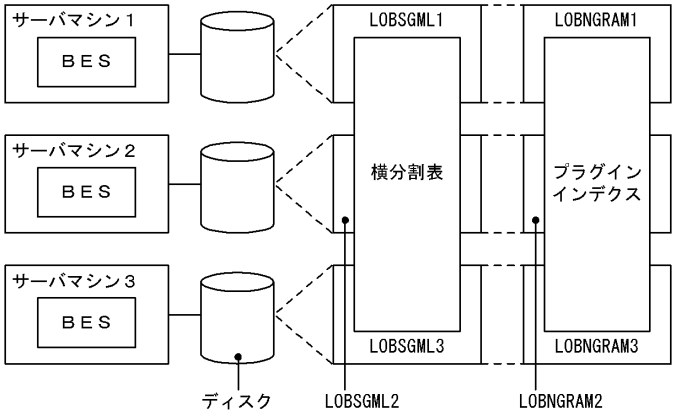
プラグインインデクスの横分割の形態を図 13-8 に示します。図 13-8 の形態でのプラグインインデクスの横分割の例を図 13-9 に示します。

**図 13-8　プラグインインデクスの横分割の形態（HiRDB/ パラレルサーバの場合）**

● バックエンドサーバ内の横分割




● バックエンドサーバ間の横分割




（凡例）
ＢＥＳ：バックエンドサーバ
LOBSGML1～LOBSGML3及び
LOBNGRAM1～LOBNGRAM3：ユーザＬＯＢ用ＲＤエリア

注　表とプラグインインデクスの対応は１対１になります。

**図 13-9 プラグインインデクスの横分割（キーレンジ分割）の例（HiRDB/ パラレルサーバの場合）**

● 薬品管理表をユーザＬＯＢ用ＲＤエリアLOBSGML1～LOBSGML3に横分割します。

LOBSGML1 薬品管理表

| 薬品ＩＤ | 取扱説明書 |
|---|---|
| 薬品１ | <添付文書データ><効能>下痢，食あたり，水あたり，・・・</効能><br><用法・用量>大人(20才以上)1回10錠，・・・食後に服用する。</用法・用量><br>:<br><使用上の注意>小児の手の届かない･･･。冷蔵庫に保管･･･。</使用上の注意></添付文書データ> |
| 薬品２ | <添付文書データ><効能>頭痛，歯痛，・・・</効能><用法・用量>大人(20才以上)1回5包・・・服用間隔は24時間以上おいてください。</用法・用量><br>:<br><使用上の注意>小児の手の届かない･･･。頭痛以外の場合は･･･。</使用上の注意><br></添付文書データ> |
| : | : |

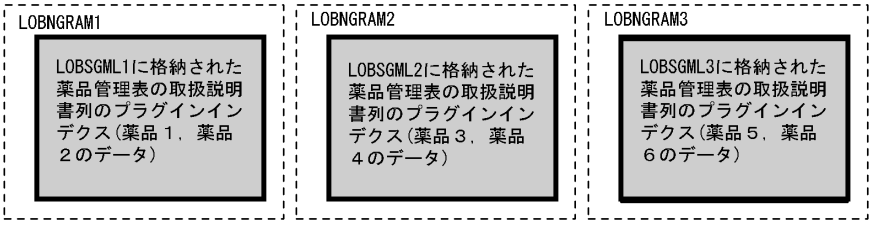
LOBSGML2 薬品管理表

| 薬品ＩＤ | 取扱説明書 |
|---|---|
| 薬品３ | <添付文書データ><効能>かぜ，かぜと思われる症状，・・・</効能><br><用法・用量>1日5回・・・食後なるべく5分以内に・・・</用法・用量><br>:<br><使用上の注意>日の当たらない場所に・・・。服用後は車等の・・・</使用上の注意></添付文書データ> |
| 薬品４ | <添付文書データ><効能>目の疲れ，肩こり，・・・</効能><br><用法・用量>成人(20才以上)1日10錠を・・・服用してください。</用法・用量><br>:<br><使用上の注意>小児の手の届かない･･･。</使用上の注意></添付文書データ> |
| : | : |

LOBSGML3 薬品管理表

| 薬品ＩＤ | 取扱説明書 |
|---|---|
| 薬品５ | <添付文書データ><効能>打撲，ねんざ，筋肉痛，・・・</効能><br><用法・用量>適当な大きさに切り，・・・１日１～２回・・・</用法・用量><br>:<br><使用上の注意>定められた使用方法を厳守してください。・・・</使用上の注意><br></添付文書データ> |
| 薬品６ | <添付文書データ><効能>すり傷，切り傷，・・・</効能><br><用法・用量>成人(20才以上)1日１回患部に噴霧・・・浸して塗布。</用法・用量><br>:<br><使用上の注意>万一目に入った場合には･･･。</使用上の注意></添付文書データ> |
| : | : |

● 横分割した薬品管理表に対応してプラグインインデクスを横分割します。




（凡例）

：プラグインインデクス

〔説明〕

取扱説明書列にプラグインインデクスが設定してあるとします。
薬品管理表を薬品 ID を条件として，ユーザ LOB 用 RD エリア LOBSGML1 ～ LOBSGML3 にわたって横分割しています。これに対応して，プラグインインデクスを LOBNGRAM1，LOBNGRAM2，LOBNGRAM3 に格納しています。

### (4) 設計上の考慮点

横分割表を格納するユーザ LOB 用 RD エリアと，横分割表のそれぞれに対応するプラグインインデクスを格納するユーザ LOB 用 RD エリアを分ける必要があります。

### (5) 注意

横分割すると RD エリアの数が増えるため，RD エリア指定のデータベースのバックアップ時には，表とインデクスの対応が 1 対 1 であることに注意してください。

# 14 RD エリアの設計

この章では，RD エリアを構成するセグメント及びページを設計する上で検討する項目について説明します。


14.1 RD エリアを設計するときの検討項目

14.2 セグメント

14.3 ページ

14.4 リスト用 RD エリアの設計

14.5 空き領域の再利用機能

14.6 共用 RD エリア（HiRDB/ パラレルサーバ限定）

# 14.1 RD エリアを設計するときの検討項目

RD エリアを構成するセグメントやページの大きさによって，ディスク所要量が異なります。この点を考慮して RD エリアを設計する必要があります。RD エリアを設計するときの検討項目を表 14-1 に，RD エリアに関する最大値・最小値を表 14-2 に示します。

**表 14-1 RD エリアを設計するときの検討項目**

| 設計作業ごとの検討項目 | | 長 所 | 短 所 | 記載箇所 |
|---|---|---|---|---|
| セグメントサイズ | 大きくした場合 | 行長が変わるような更新をする場合や，クラスタキーを指定した表に行を追加する場合，行を格納した特定のページに近接する未使用のページを確保できるため，データの入出力時間を削減できます。 | セグメント数が少なくなるため，一つのユーザ用 RD エリアに格納できる表とインデクスの数が少なくなります。 | 14.2.1 |
| | 小さくした場合 | 一つのユーザ用 RD エリアにデータ量の少ない表を多く格納する場合は，余分な未使用ページを削減できます。 | • ユーザ用 RD エリアに大量のデータを追加すると，セグメントの割り当て回数が増加するため，オーバヘッドが大きくなります。<br>• セグメントの数が多くなるため，表を削除したり，表の全行削除をしたりする場合，排他制御の資源が多くなります。 | |
| セグメント内の空きページ比率 | 設定した場合 | 表にデータを追加する場合，クラスタキーを指定している表に対しては，クラスタキーの値に近い場所のページにデータを格納できるため，データの入出力回数を削減できます。 | 設定値を大きくする分，ディスク所要量が増加します。 | 14.2.2 |
| | 0 にした場合 | ディスク所要量を削減できます。 | クラスタキーを指定している表の場合，データの追加時にクラスタキーの値に近い場所にデータを格納できなくなるため，格納状態が乱れ，データの入出力回数の削減効果がなくなります。 | |
| ページ長 | ページ内の未使用領域の比率を設定した場合 | • UPDATE 文で行長が元の行よりも長くなる更新をしても，連続した空き領域が更新後の行長よりも大きい場合は，該当する行がそのページに収まるようになります。<br>• INSERT 文で行を繰り返し追加するときに，クラスタキーの値に近い場所のページに，1 ページ内が一杯になるまで行を追加できるようになります。 | FIX 属性の表の場合は格納効率が悪くなります。 | 14.3.2 |
| | ページ内の未使用領域の比率を 0 にした場合 | FIX 属性の表の場合，データが昇順になるようなときは，格納効率が良くなります。 | 行長が元の行よりも長くなる更新をすると，行が複数ページにわたるため，行にアクセスするときにオーバヘッドが掛かります。 | |

| 設計作業ごとの検討項目 | | 長　所 | 短　所 | 記載箇所 |
|---|---|---|---|---|
| 空き領域の再利用 | 使用した場合 | • 使用中セグメント内の空き領域を有効活用できます。<br>• RD エリア満杯後の空き領域サーチの性能が向上します。 | 再利用するのに十分空き領域がない場合，空き領域サーチのオーバヘッドが増えます。 | 14.5 |
| | 使用しない場合 | 空き領域が十分ある状態であれば，挿入処理が高速にできます。 | RD エリアの格納効率が低下します。また，RD エリア満杯後の空き領域サーチの性能が低下します。 | |
| 共用 RD エリア | 使用した場合 | アクセスが多いが，分割が難しい表を共用 RD エリアに格納すると，全バックエンドサーバから参照できるため，並列処理の効率が上がります。 | 共用表を更新する場合，更新する共用表がある共用 RD エリアに排他を掛けるので，共用 RD エリアのほかの表にアクセスする業務があると，デッドロックが発生することがあります。 | 14.6 |
| | 使用しない場合 | 共用 RD エリアが原因となるデッドロックや，サーバ間のグローバルデッドロックが発生しません。 | 結合処理などの複雑な検索処理の場合，複数のバックエンドサーバ接続とデータ転送のオーバヘッドが増えます。 | |

**表 14-2　RD エリアに関する最大値・最小値**

| 項目 | 最大値・最小値 |
|---|---|
| 総 RD エリア数 | 3 ～ 8,388,592 |
| マスタディレクトリ用 RD エリア数 | 1 |
| データディレクトリ用 RD エリア数 | 1 |
| データディクショナリ用 RD エリア数 | 1 ～ 41 |
| ユーザ用 RD エリア数 | 1 ～ 8,388,589 |
| データディクショナリ LOB 用 RD エリア数 | 1 ～ 2 |
| ユーザ LOB 用 RD エリア数 | 0 ～ 8,388,325 |
| レジストリ用 RD エリア数 | 0 ～ 1 |
| レジストリ LOB 用 RD エリア数 | 0 ～ 1 |
| リスト用 RD エリア数 | 0 ～ 8,388,588 |
| 1RD エリア中の HiRDB ファイル数 | 1 ～ 16 |
| 1RD エリア中の実表数 | 0 ～ 500 |
| 1RD エリア中のインデクス数 | 0 ～ 500 |
| 1RD エリア中のリスト数 | 0 ～ 50,000 |
| 総 HiRDB ファイル数 | 1 ～ 134,217,728 |

**●インデクス格納 RD エリアの容量見積もりについて**

インデクス格納 RD エリアの容量見積もりについては，「16.1　ユーザ用 RD エリアの容量の見積もり」を参照してください。ここでは，容量見積もり時の注意事項を説明します。

1. データベース作成ユティリティ又はデータベース再編成ユティリティでインデクスを一括作成した直後は，データはきれいに格納されています。その後のデータ挿入時にすべてのキーを昇順に挿入しないかぎり，インデクスページスプリットが発生するため，インデクスの一括作成時よりインデクスの容量が大きくなります。
2. インデクスページは基本的に使用中空きページを再使用しません。したがって，キー値を変更する

ような更新及び削除を行った場合，変更又は削除前のキーが格納されていたページを再使用できません。このように再使用されないむだな使用中空きページができてしまいます。ただし，使用中空きページを再利用する運用もできます。詳細は，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

3. キー値の重複がある場合とない場合では基本的にインデクス構造が異なります。このため，正しく重複数を求めないとインデクスの容量が大きく異なってしまいます。インデクスレコード件数が少ない場合は，インデクスの容量に比べてこの誤差の占める割合が大きくなります。

# 14.2 セグメント

セグメントには次の表に示す状態があります。

**表 14-3 セグメントの状態**

| セグメントの状態 | 説明 |
|---|---|
| 使用中セグメント※ | 表又はインデクスのデータを格納しているセグメントです。特に，データが満杯でセグメント内にデータを追加できないセグメントを**満杯セグメント**といい，データの削除でセグメント内の全ページが空きページ（使用中空きページ又は未使用ページ）のセグメントを**使用中空きセグメント**といいます。 |
| 未使用セグメント | 使用されたことがないセグメントです。このセグメントは RD エリア内のすべての表（又はインデクス）が使用できます。 |
| 空きセグメント | データを格納していないセグメントです。使用中空きセグメントと未使用セグメントは空きセグメントになります。 |

注※

使用中セグメントを使用できるのは，このセグメントにデータを格納した表又はインデクスだけです。ほかの表又はインデクスはこのセグメントを使用できません。

## 14.2.1 セグメントサイズの決定

通常，RD エリアのセグメントサイズは，RD エリア格納ページ数の 1/10 程度にすることをお勧めします。ただし，セグメントサイズの最大は 16,000 ページであるため，RD エリアの容量が大きい場合は，1/10 以下になります。

セグメントサイズの大小による効果と注意を次に示します。

### (1) セグメントサイズを大きくした場合

**性能の向上**

- 行長が変わるような更新をする場合や，クラスタキーを指定した表に行を追加する場合，行を格納した特定のページに近接する未使用のページを確保できるため，データの入出力時間を削減できます。
- 同じ表のデータが連続したページに格納されるため，**プリフェッチ機能**による一括入力の効果が得られます。プリフェッチ機能を使用する場合，セグメントサイズは，システム共通定義の pdbuffer オペランドの -p オプションで指定する一括入力最大ページ数に合わせることをお勧めします。

**注意事項**

- セグメント数が少なくなるため，一つのユーザ用 RD エリアに格納できる表とインデクスの数が少なくなります。

### (2) セグメントサイズを小さくした場合

**ディスク所要量の削減**

- 一つのユーザ用 RD エリアにデータ量の少ない表を多く格納する場合は，余分な未使用ページを削減できます。

**注意事項**

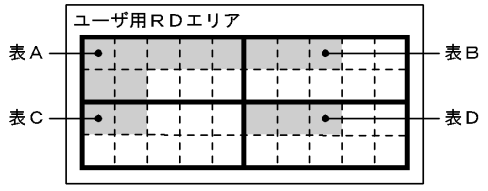
- セグメントサイズを小さくしたユーザ用 RD エリアに大量のデータを追加すると，セグメントの割り当て回数が増加するため，オーバヘッドが大きくなります。
- セグメントの数が多くなるため，表を削除したり，表の全行削除をしたりする場合，排他制御の資

源が多くなります。

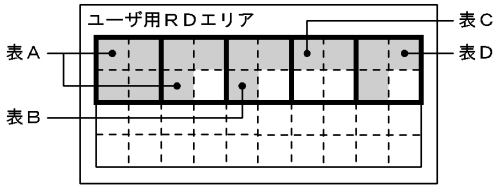
セグメントサイズの大小によるユーザ用 RD エリアの概要を次の図に示します。

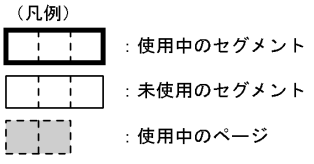
**図 14-1 セグメントサイズの大小によるユーザ用 RD エリアの概要**

● セグメントサイズが大きい場合




● セグメントサイズが小さい場合



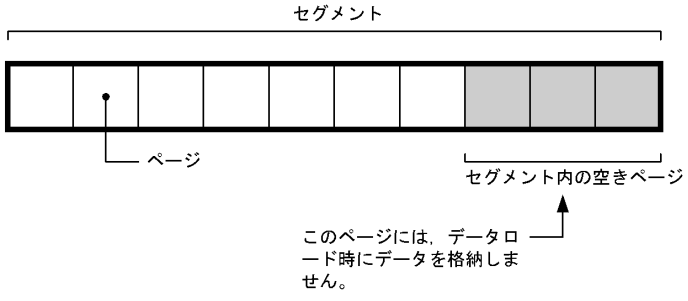

### (3) 設定方法

セグメントサイズは，データベース初期設定ユティリティ（pdinit）又はデータベース構成変更ユティリティ（pdmod）の create rdarea 文で設定します。

## 14.2.2 セグメント内の空きページ比率の設定

表を定義するときにセグメント内に割り当てる空きページの割合を**セグメント内の空きページ比率**といいます。ここでいう空きページとは，未使用ページのことです。セグメント内の空きページの概要を次の図に示します。

**図 14-2 セグメント内の空きページの概要**



### (1) 設定の効果

セグメント内の空きページ比率の設定の効果を次に示します。

**性能の向上**

- 表にデータを追加する場合，クラスタキーを指定している表に対しては，クラスタキーの値に近い場所のページにデータが格納されます。このため，データの入出力回数を削減できます。

### (2) 適用基準

- クラスタキーを指定した表に，データベース作成ユティリティ（pdload）などでデータを格納した後，データの追加が大量に発生するような場合は，セグメント内の空きページ比率を設定するようにします。
- データの追加又は更新がほとんど発生しない表の場合は，セグメント内の空きページ比率を 0 にします。

### (3) 設定方法

セグメント内の空きページ比率は，定義系 SQL の CREATE TABLE の PCTFREE オプションで指定します。

### (4) 注意

セグメント内の空きページ比率に 0 を指定した場合，クラスタキーを指定している表にデータを追加するときは，クラスタキーの値に近い場所にデータを格納できなくなります。このため，データの格納状態が乱れ，データの入出力回数の削減効果がなくなります。

## 14.2.3 セグメントの確保と解放

表を定義したときにはセグメントを確保しません。表にデータを格納するときに必要に応じてセグメントを確保します。一度確保したセグメント（一度使用したセグメント）はそのセグメントを解放しないかぎり，ほかの表又はインデクスが使用できません。このため，データの追加と削除を繰り返した場合，データ量が増えていないのに RD エリアが容量不足になることがあります。これを防ぐには次に示す操作を定期的に行ってセグメントを解放してください。

- データベース再編成ユティリティ（pdrorg コマンド）による表の再編成又はインデクスの再編成
- 空きページ解放ユティリティ（pdreclaim コマンド）による使用中空きセグメントの解放

表の再編成，インデクスの再編成，使用中空きセグメントの解放については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

なお，これらの操作以外にも次に示す操作をした場合はセグメントを解放します。

- PURGE TABLE 文の実行
- RD エリアの再初期化
- 表の定義の削除
- インデクスの定義の削除
- データロードを作成モード（-d オプション指定）で実行

# 14.3 ページ

ページには次の表に示す状態があります。

**表 14-4 ページの状態**

| ページの状態 | 説明 |
|---|---|
| 未使用ページ | まだ割り当てられていないページです。 |
| 使用中空きページ | データの削除※によって，データが格納されていないページです。 |
| 使用中ページ | データが格納されていて，データを追加できる空き領域があるページです。<br>空き領域の再利用機能を使用している表の場合は，データの削除※によってページ内の空き領域が使用できないために，データを追加できなかったページを含みます。 |
| 使用中満杯ページ | データが格納されていて，データを追加できる空き領域がないページです。<br>空き領域の再利用機能を使用していない表及びインデクスの場合は，データの削除※によってページ内の空き領域が使用できないために，データを追加できなかったページを含みます。 |

注※

データの削除を実行したトランザクションが COMMIT するまで，データの削除によって発生した空き領域は使用できません。

## 14.3.1 ページ長の決定

### (1) ページ長を決定するときの考慮点

ページ長を決定するときの考慮点を次に示します。

1. 全件検索及び更新，大量検索及び更新をする業務で，次に示す条件を満たす表，インデクスを格納するページ長は大きくします。
   - インデクスを付けない表を格納する RD エリア
   - クラスタキーを指定した表及びそのインデクスを格納する RD エリア
   - 大量検索，大量更新の範囲条件となるインデクスを格納する RD エリア
2. ページ長は，RD エリアに格納する表の行長を基に，できるだけ無効領域が作られない長さにします。

   **無効領域＝** MAX（mod（**(ページ長－** 48**)，(行長＋** 2**)**)**，ページ長－** 48 **－（行長＋** 2**）×** 255**）**
3. ページ内の未使用領域の比率を指定する場合は，次に示す計算式を目安にします。

   **（ページ長×ページ内の未使用領域の比率）÷** 100 **－行長×ページ内の未使用領域に格納できる行数**

   この場合，ページ内の未使用領域に 1 行も格納できなくなるような無意味なページ内の未使用領域の比率は指定しないようにします。
4. インデクスを格納するページの場合は，一般的に 4096 ～ 8192 バイト程度が，入出力効率の面から適しています。
5. 列のデータ型が VARCHAR，NVARCHAR 及び MVARCHAR で，定義長が 256 バイト以上の場合は，該当する列のデータが別のページに分岐されます。また，256 バイト以上の可変長文字列データがある場合，その平均長以上で最も小さいページ長にします。
6. 列のデータ型が VARCHAR，NVARCHAR 及び MVARCHAR の場合，INSERT 文で行をナル値として挿入すると，UPDATE 文で該当する列のデータを実データに更新するときに，更新したデータの長さによっては，該当する列のデータが別のページに分岐されることがあります。可変長文字列データの初期値をナル値として，後から実データに更新することが多い場合は，更新後の行長を見込んでページ長を決定します。

7. HiRDB では，ページ又は行単位に排他制御ができます。行レベルの排他制御をする場合には，1 ページに格納できる行数ができるだけ多くなるように，行長に応じてページ長を設定します。このとき，次に示す点を考慮します。
   - ページ内の未使用領域の比率を最低限にします。
   - ページの入出力要求に対するグローバルバッファの排他待ちが少なくなるようにします。更新が多い表の場合，排他待ちの回数が増える可能性があるため，ページ長を小さくします。
   - ページの入出力要求に対するページの入出力待ちが少なくなるようにします。ランダムアクセスが中心の業務で，ページ長が大き過ぎると，アクセス単位である行長に対する実際の入出力単位が大きくなって不要なデータ転送が発生するため，ページ長を小さくします。
     ただし，UPDATE 文で列のデータ型が VARCHAR，NVARCHAR 及び MVARCHAR のデータに対して行長が変更されるような更新を頻繁にする場合は，表を定義するときにページ内の未使用領域の比率を高めに設定します。ページ内の未使用領域の比率の設定については，「14.3.2　ページ内の未使用領域の比率の設定」を参照してください。

#### (2) 設定方法

ページ長は，データベース初期設定ユティリティ（pdinit）又はデータベース構成変更ユティリティ（pdmod）の create rdarea 文で設定します。

#### (3) ページ長を決定するときの注意

表に行を追加する場合，実際の行の長さ（データ型が VARCHAR，NVARCHAR 及び MVARCHAR の列を除いた長さ）がページ長を超えると，エラーになります。実際の行の長さは，「16. RD エリアの容量の見積もり」に記載されているディスク所要量の見積もり式を参照して求めます。求めた行の長さが，使用するユーザ用 RD エリアのページ長より大きい場合は，ユーザ用 RD エリアを再初期化して，ページ長を再設定する必要があります。RD エリアの再初期化は，データベース構成変更ユティリティ（pdmod）で実行します。RD エリアの再初期化については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

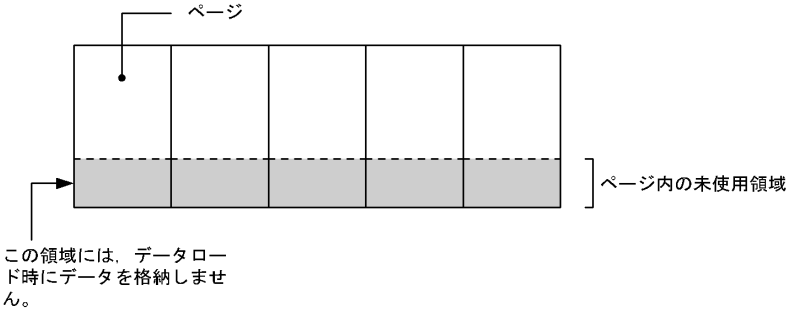
### 14.3.2　ページ内の未使用領域の比率の設定

表とインデクスを定義するときに，ページ内に割り当てる未使用領域の割合を**ページ内の未使用領域の比率**といいます。未使用領域を設定しておくと，データベース作成ユティリティ（pdload）又はデータベース再編成ユティリティ（pdrorg）でデータを格納するとき，未使用領域にはデータは格納されません。

ただし，データベース作成ユティリティ（pdload）で -y オプション又は option 文を指定して実行し，新しいページが確保できなかった場合は，未使用領域にもデータが格納されます。

ページ内の未使用領域の概要を次の図に示します。

**図 14-3 ページ内の未使用領域の概要**




## (1) 設定の効果

- UPDATE 文で行長が元の行より長くなるような更新をしても，連続した空き領域が更新後の行長よりも大きい場合は，該当する行がそのページに収まるようになります。
- INSERT 文で行を繰り返し追加するときに，クラスタキーの値に近い場所のページに，1 ページ内が一杯になるまで行を追加できるようになります。

## (2) 適用基準

1. クラスタキーを指定した表に行を追加する場合に設定します。
2. FIX 属性の表の場合，データが昇順になるようなときは，ページ内の未使用領域の比率を 0 にした方が，格納効率が良くなります。
3. 行長が元の行よりも長くなるような更新をする場合に設定します。
4. 行長が元の行長より長くなるのは，次に示す更新をしたときです。

- ナル値から実データに更新したとき
- データ型が VARCHAR，NVARCHAR，MVARCHAR，及び BINARY の列を長い値になるように更新したとき

## (3) 設定方法

ページ内の未使用領域の比率は，定義系 SQL の CREATE TABLE 又は CREATE INDEX の PCTFREE オプションで指定します。

## (4) 注意

設定した未使用領域に余裕がない場合は，行長を元の行長より長くなるような更新をしたときに，1 行が複数ページにわたるため，入出力回数が増えます。

## (5) ページ内の未使用領域の比率の求め方

● 最初に格納した行の行長が L1 で，最終的に L2 になる場合は，次に示す計算式で求めた値を未使用領域の比率とするのが一般的です。

**ページ内の未使用領域の比率** =（(L2-L1）÷ L2）× 100（%）

● 表にクラスタキーを指定している場合は，次のように求めます。

1. データベース作成ユティリティ（pdload）でこの表にデータを格納する件数をページ当たりで求めます。これを m 件とします。
2. その後，更に何件のデータを格納するかを求めます。これを n 件とします。
3. 1. と 2. で求めた m と n から，次に示す計算式でページ内の未使用領域の比率を求めます。

**ページ内の未使用領域の比率** =（n ÷（m ＋ n））× 100（%）

## 14.3.3 ページの確保と解放

### (1) ページの確保

表を定義したときにはページを確保しません。表にデータを格納するときに必要に応じてページを確保します。一度確保したページ（一度使用したページ）はそのページを解放しないかぎり，再使用できません。

インデクスを定義した場合は，データ件数に応じてページを確保します。データ件数 0 件の場合は 1 ページ（ルートページ）だけ確保します。ただし，CREATE INDEX に EMPTY オプションを指定した場合（インデクスの実体を作成しない場合）はページを確保しません。

**注意事項**

1. 非 FIX 表で行長が変わるデータ更新を行った場合は，行長が減った分の領域を再使用できません。
2. インデクスページは削除ページに格納されていたキー値と同じキー値が追加されないかぎり，そのページを再使用しません。
3. データの削除によって発生した空き領域があるページを再使用するときは次に示す制限があるので注意してください。

- 256 バイト以上の VARCHAR，BINARY 型，抽象データ型，及び繰返し列の分岐行は，そのページを使用できません。
- セグメントの使用率が 100%になるまで，データの挿入時にそのページを使用できません。
- DELETE を実行したトランザクションが COMMIT を発行するまで，DELETE によって発生した空き領域を使用できません。

### (2) ページの解放

- セグメントが解放されるとセグメント内のページは解放されます。
- EXCLISIVE 指定の LOCK 文で排他を掛けた表に対して，UAP がページ内の全行を削除すると，ページを解放します。インデクスのページは解放されません。
- PURGE TABLE 文を実行すると，表及びインデクスのページとセグメントが解放されます。ただし，インデクスのルートページは残ります。
- 空きページ解放ユティリティ（pdreclaim コマンド）で使用中空きページを解放できます。使用中空きページの解放については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

# 14.4 リスト用 RD エリアの設計

## (1) リスト用 RD エリアの必要数

1 リスト用 RD エリアに作成できるリストの最大数は，次に示すオペランドで指定します。

- データベース初期設定ユティリティ（pdinit）の create rdarea 文の max entries オペランド
- データベース構成変更ユティリティ（pdmod）の create rdarea 文の max entries オペランド
- データベース構成変更ユティリティ（pdmod）の initialize rdarea 文の max entries オペランド

指定できる最大数は 500 ～ 50000 です。

## (2) ページ長，セグメントサイズの求め方

リストにはリストの基表の行識別子が格納されます。表のように直接データを格納しないので，1 ページ内に比較的大量の行を格納できます。したがって，ページ長及びセグメントサイズをリストに格納する行数に比べて大きく設定した場合，RD エリア内に余分な空き領域が発生するので注意してください。

リスト用 RD エリアのページ長，セグメントサイズを決定する場合，あらかじめサーバ内に作成するリストの平均行数をおおよそ見積もり，次に示すどれかのケースに基づいてページ長及びセグメントサイズを設定してください。

| 条件 | ページ長 | セグメントサイズ |
|---|---|---|
| サーバ内に作成するリストの平均行数が 3000 行未満の場合 | 4096 | 1 |
| サーバ内に作成するリストの平均行数が 3000 行～ 6000 行の場合 | 4096 | 2 |
| サーバ内に作成するリストの平均行数が 6000 行を超える場合 | （a）参照 | （b）参照 |

### (a) リストの平均行数が 6000 行を超える場合のページ長の求め方

ページ長は，通常 4096 ～ 8192 バイトの範囲にしてください。ただし，リストの入出力回数を減らしてリストの入出力時間を短縮したい場合は，ページ長を大きくしてもかまいません。ただし，ページ長を大きくすると，グローバルバッファの必要量も大きくなるため，共用メモリが大量に必要となるので注意してください。

なお，ページ長は次に示す計算式を満たす値にしてください。

**計算式**

リストの 1 ページに格納できる行数≦ サーバ内に作成するリストの平均行数÷ 2

リストの 1 ページに格納できる行数は，次に示す計算式から求めます。

**リストの 1 ページに格納できる行数＝↓｛ページ長− 70 −（a × 8)｝÷ 4 ↓**

a：サーバ内のリストの基表を格納している RD エリアの HiRDB ファイルの総数

### (b) リストの平均行数が 6000 行を超える場合のセグメントサイズの求め方

セグメントサイズは 1 リストへの RD エリア内領域の割り当て単位です。したがって，1 セグメントが 1 リストへの最低割り当てサイズとなります。セグメントサイズの目安を次に示します。

- セグメント割り当てのオーバヘッドを削減したい場合，セグメントサイズを大きくしてください。

- リスト用 RD エリアのグローバルバッファにプリフェッチ機能を指定する場合は，セグメントサイズに 2 以上の値を指定してください。指定しないと，プリフェッチ機能が動作しないので注意してください。
- セグメントサイズを大きくすると，セグメント内に余分な未使用ページが発生する可能性が高くなります。余分な未使用ページを減らしたい場合は，セグメントサイズを小さくしてください。
- セグメントサイズは次に示す計算式を満たす値にしてください。

**リストの 1 セグメントに格納できる行数**
**≦サーバ内に作成するリストの平均行数÷ 2**

リストの 1 セグメントに格納できる行数は，次に示す計算式から求めます。

**リストの 1 セグメントに格納できる行数**
**＝リストの 1 ページに格納できる行数×セグメントサイズ**

### (3) セグメント数の求め方

リスト用 RD エリアに必要なセグメント数は，次に示す計算式から求めます。

**計算式**

リスト用 RD エリアに必要なセグメント数＝↑｛↑ a ÷ b ↑×（c ＋ 0.5)｝↑

a：サーバ内のリストの数

b：サーバ内のリスト用 RD エリアの数

c：1 リストが使用するセグメント数の平均値
次に示す計算式から求めます。
**↑サーバ内のリストの平均行数÷リストの 1 セグメントに格納できる行数↑**

セグメント数が不足すると，リストが作成できなくなるため，実際には上記の計算式で見積もったセグメント数に余裕を持たせてください。

# 14.5 空き領域の再利用機能

空き領域の再利用機能を使用すると，データの削除でできる空き領域を有効活用できます。ここでは，次の項目について説明します。

- データ格納時のサーチ方式
- 空き領域の再利用機能とは
- 効果と適用基準
- 使用前の考慮点
- 環境設定
- 実行状態の確認
- 注意事項

## 14.5.1 データ格納時のサーチ方式

表にデータを格納するとき，格納領域をサーチする方式には次の二つのページサーチモードがあります。

**● 新規ページ追加モード**
使用中セグメントの最終ページが満杯になると，新規に未使用セグメントを確保します。RD エリア中に未使用ページがなくなると，使用中ページの空き領域を使用中セグメントの先頭からサーチして空き領域にデータを格納します。

**● 空きページ再利用モード**
使用中セグメントの最終ページが満杯になると，未使用セグメントを確保する前に使用中セグメント内の使用中ページの空き領域をサーチします。また，次回サーチ開始位置を記憶し，次に空き領域をサーチするときそこからサーチを開始します。

## 14.5.2 空き領域の再利用機能とは

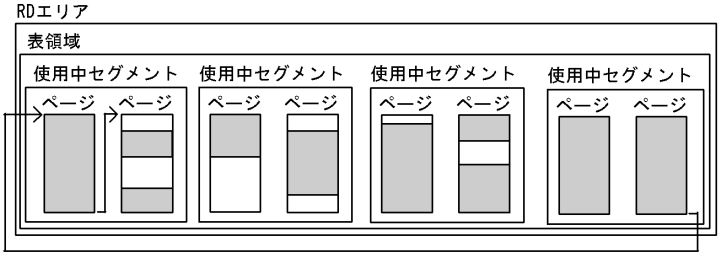
**空き領域の再利用機能**とは，表の使用中セグメントがユーザの指定したセグメント数に達し，そのセグメントが満杯になるとページサーチモードを空きページ再利用モードに切り替えて，使用中ページの空き領域を使用する機能です。指定した数のすべてのセグメントに空き領域がなくなると，新規ページ追加モードに切り替わり，新規に未使用セグメントを確保します。

なお，セグメント数を指定しないと RD エリア中に未使用ページがなくなるまで空き領域を再利用しません。空き領域の再利用機能を使用しない場合，毎回使用中セグメントの先頭から空き領域をサーチします。この機能を使用している場合，空きページ再利用モードに切り替わった後，次回サーチ位置を記憶してそれ以降をサーチするのでこの機能を使用しない場合より効率良くサーチできます。

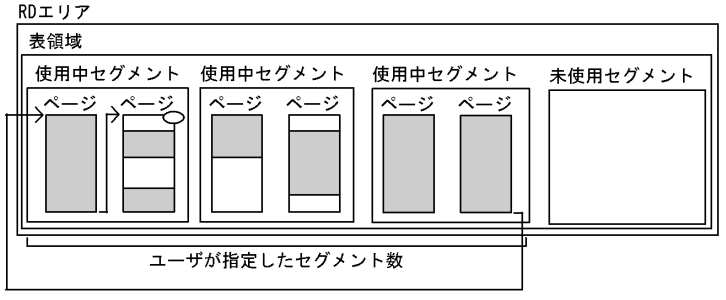
空き領域の再利用機能の概要を次の図に示します。

**図 14-4　空き領域の再利用機能の概要**

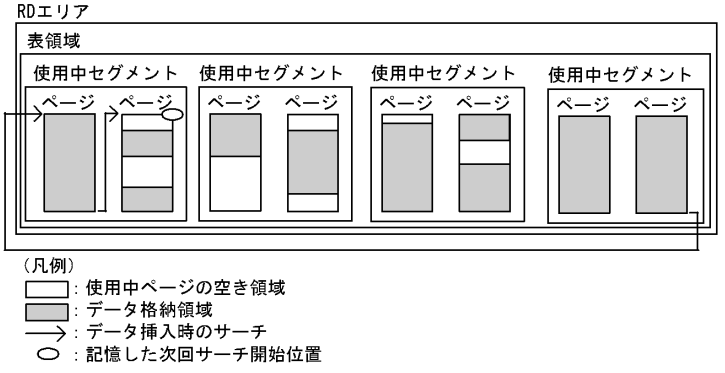
●空き領域の再利用機能を使用しない場合




●空き領域の再利用機能を使用した場合（セグメント数指定あり）




●空き領域の再利用機能を使用した場合（セグメント数指定なし）




〔説明〕

- 空き領域の再利用機能を使用しない場合
  RD エリア中に未使用ページがなくなると，その後データが挿入されるたびに使用中セグメントの先頭から使用中ページの空き領域をサーチしてそこに空き領域にデータを格納します。
- 空き領域の再利用機能を使用した場合（セグメント数指定あり）
  指定したセグメント数に達した後で表にデータを挿入しようとすると，未使用セグメントを確保しないで，使用中ページの空き領域を使用中セグメントの先頭からサーチしてそこにデータを格納します。そこで次回サーチ開始位置を記憶しておき，次に空き領域をサーチするときはそこからサーチを開始します。
- 空き領域の再利用機能を使用した場合（セグメント数指定なし）
  RD エリア中に未使用ページがなくなってからデータを挿入しようとすると，使用中ページの空き領域を使用中セグメントの先頭からサーチしてそこにデータを格納します。そこで次回サーチ開始位置を記憶しておき，次に空き領域をサーチするときはそこからサーチを開始します。

## 14.5.3 効果と適用基準

### (1) 効果

この機能を使用すると，次の効果が期待できます。

- 空き領域の有効活用
  使用中ページの空き領域を再利用するため，最小限の RD エリア容量で運用でき，データベースの再編成回数を減らせます。また，1RD エリアに複数の表及びインデクスを格納する場合に，ある表に対して集中して挿入及び削除処理が実行された場合の領域占有を回避できます。
- 可変長列及び BINARY 型の列のページ不足エラーの回避
  通常，ノースプリットオプションを指定しないで 256 バイト以上の可変長文字列を挿入したり，1 ページに入らない BINARY 型の列を挿入したりすると，未使用ページが確保されます。使用中空きページがあっても，未使用ページが確保できなければエラーになりますが，空き領域の再利用機能を使用すると使用中空きページを未使用ページの代わりに確保するのでページ不足エラーを回避できます。
- 使用中ページの空き領域サーチ時のオーバヘッドの削減
  空きページ再利用モードの場合，次回サーチ開始位置を記憶し，そこからサーチを開始するのでオーバヘッドが削減されるため，高速に処理できます。

### (2) 適用基準

- 削除と挿入を繰り返すため，データ量に対してセグメントが大量に消費され，頻繁に再編成しなければならない業務で，再編成の回数をできるだけ減らしたい場合に空き領域の再利用機能を使用してください。この機能の使用をお勧めする場合の業務特性と効果を次に示します。
  - 削除（更新），挿入を含み，データ量の増加がない場合
    空き領域の再利用機能で格納データの最大サイズを指定しておけば，その後は削除されるデータの領域を優先して再利用するので，新規領域を追加しないで業務を継続でき，再編成が不要になります。

    **（例）**行政電子窓口
    電子窓口で受け付け業務を 24 時間するシステムで，申請受け付け時に申請データを挿入し，保管期限経過後に削除します。最大保管期限内に受け付ける申請データ数のセグメントサイズを指定すれば，削除されるデータの領域を再利用するので，新規領域を追加しないで業務を継続できます。そのため，再編成が不要となり，業務を停止することなく 24 時間サービスができます。

  - 削除（更新），挿入を含み，データ量が徐々に増加する場合
    徐々に増加するデータを新規領域だけでなく，削除した領域にも格納するため，格納効率が向上します。

    **（例）**顧客管理
    新規顧客のデータを挿入し，不要になった顧客登録データを削除します。初期顧客データ登録終了後，追加，削除業務開始前のセグメントサイズを指定すれば，その後は削除された顧客データの領域を再利用しながら追加されます。

- 挿入処理は新規に未使用ページや未使用セグメントにデータを格納する方が性能が良くなります。そのため，短い周期でデータベース再編成ユティリティ（pdrorg）を実行できる場合は，空き領域の再利用機能を適用しないでデータベースを再編成する方が性能が良くなります。

## 14.5.4 使用前の考慮点

空き領域の再利用機能が有効になるのは，削除処理によって空き領域が十分にできる場合です。空き領域が十分にないときや全くないときに空き領域のサーチをするなど，むだに空き領域サーチをする場合，セ

グメント内のページ数の指定を大きくしたり，この機能を中止する必要があります。セグメント内のページ数指定を変更するためには，RD エリアを再作成（削除又は追加）する必要があるため，最初の設計時に指定するセグメント数，セグメントサイズは十分に考慮してください。

● 次の場合，SEGMENT REUSE オプションでセグメント数は省略してかまいません。
- RD エリア内に表が一つで，インデクス混在なし，自動増分指定なし

● 次の場合，SEGMENT REUSE オプションでセグメント数を指定する必要があります。
- RD エリア内に表が一つで，インデクスは混在なし，自動増分指定あり
- RD エリア内に表が一つで，インデクスは混在あり
- RD エリア内に表が複数あり

データ件数が増加する場合，セグメント数を指定し，1 セグメントが満杯になるまでに十分削除がされるようなセグメントサイズにしてください。データ件数が増加しない場合，表が使用する総セグメント数を見積もって，セグメント数を指定すればセグメントサイズは考慮不要です。ただし，同一 RD エリア内の表で，再利用するセグメント数の合計（インデクスが混在している場合，同一 RD エリア内の表で再利用するセグメント数とインデクスの見積もりセグメント数の合計）は RD エリアの総セグメント数以下にしてください。

なお，自動増分指定された表に空き領域の再利用機能が使用された場合，領域の増分を優先し，増分した領域が指定されたセグメント数になると，空き領域の再利用を実行します。

### 14.5.5 環境設定

空き領域の再利用機能を使用するための環境設定について次に示します。

1. pd_assurance_table_no オペランドに空き領域の再利用機能を使用する表数を指定します。
   分割表の場合は 1 分割を 1 表として計算します。HiRDB/ パラレルサーバの場合，バックエンドサーバごとに計算し，その最大数をこのオペランドに登録します。
   なお，CREATE TABLE で定義又は ALTER TABLE で定義変更した表は，pd_assurance_table_no オペランドの指定数（予約数）まで空き領域の再利用機能を使用できます。予約数を超えた表に挿入が実行された場合，KFPH22030-W メッセージが出力され，空き領域の再利用機能は適用されません。この場合，pd_assurance_table_no オペランドの指定値を増やすと定義したすべての表に空き領域の再利用機能が適用されます。ALTER TABLE の ADD RDAREA 指定で表格納 RD エリアを追加して定義数が予約数を超えた場合や，HiRDB/ パラレルサーバで定義数が予約数を超えた場合は，空き領域の再利用を定義した分割表で RD エリアごとに空き領域の再利用が適用されたり，されなかったりする場合があります。
2. 空き領域を再利用するセグメント数を見積もり（表の総データ数から総セグメント数を見積もります。「16.1　ユーザ用 RD エリアの容量の見積もり」を参照してください），見積もったセグメント数を定義系 SQL の CREATE TABLE の SEGMENT REUSE オプションで指定します。作成済みの表に対しては ALTER TABLE の SEGMENT REUSE オプションで指定します。ここで指定したセグメント数はすべての RD エリアに適用されます。
3. 一度定義したセグメント数を変更する場合，ALTER TABLE の SEGMENT REUSE オプションで再度セグメント数を指定します。ページサーチモードとセグメント数の指定値によって，HiRDB は次のように処理します。
   - 新規ページ追加モード時
     指定されたセグメント数が使用中セグメント数より少ない場合，最後に確保したセグメント内に空き領域がなくなった後で空き領域の再利用を実行します。
   - 空きページ再利用モード時
     使用中セグメント数以下のセグメント数が指定された場合，そのまま続行します。使用中セグメント

数より多いセグメント数が指定された場合，空き領域をすべて使用した後で空き領域の再利用をいったん中止し，新規に未使用セグメントを確保します。

4. バッチ処理などで一時的に大量追加をする場合など，空き領域の再利用機能を一時中止したい場合，ALTER TABLE で SEGMENT REUSE NO を指定します。実行するとすぐに空き領域の再利用機能は中止され，新規に未使用セグメントが確保されます。
5. 空き領域の再利用機能を使用している表がセグメント確保時に出力する，RD エリアのセグメント使用率通知メッセージ（KFPH00211-I，又は KFPA12300-I）を抑止したい場合，pd_rdarea_warning_point_msgout オペランドに N を指定します。
   削除（更新），挿入を含み，かつデータ量の増加がない場合，空き領域の再利用機能を使用すると，表の再編成や RD エリアの拡張をする必要がなくなります。そのため，ユーザは RD エリアのセグメント使用率通知メッセージの出力を監視する必要もなくなります。削除（更新），挿入を含み，データ量の増加がなく，かつ次のすべての条件に該当するときは，RD エリアのセグメント使用率通知メッセージの出力を抑止できます。
   - 格納 RD エリアに，空き領域の再利用機能を使用している表だけを定義している。
   - FIX 属性の表である，又は可変長の列を含まない表である（データ長が長くなるような更新をしない）。

   ただし，次に示す場合は空き領域の再利用機能が動作しないおそれがあるため，RD エリアのセグメント使用率通知メッセージを出力し，監視を行ってください。RD エリアの使用状況に応じてユーザが対処する必要があります。
   - 空き領域の再利用機能を定義している表数が，pd_assurance_table_no オペランドで指定した予約数より多い。
   - 格納 RD エリアに対して空き領域の再利用機能を使用している表を複数定義していて，表定義の SEGMENT REUSE のセグメント数に，最大データ量サイズ以上を指定していない。

## 14.5.6 実行状態の確認

空き領域の再利用機能が有効かどうかを，データベース状態解析ユティリティ，統計解析ユティリティ，及び UAP 統計レポート機能で表示される項目から確認できます。また，空回りした場合，表（分割表の場合は分割 RD エリア）ごとの一回目にメッセージログに KFPH22031-W が出力されます。表示される項目とその説明を次に示します。

| 項目 | 説明 | 対処 |
|---|---|---|
| ページサーチモードの切り替え回数 | 新規ページ追加モードから空きページ再利用モードへ，又は空きページ再利用モードから新規ページ追加モードへのサーチモード切り替え回数です。再利用と未使用セグメントの確保が頻繁に切り替わっているということは，削除でできる空き領域より追加する領域が大きく，セグメントサイズ（ページ数）が小さいということになります。 | セグメントサイズ変更や削除実行のタイミングを見直してください。 |
| 空き領域の再利用機能のページサーチ空回り回数 | 使用中セグメント数が指定されたセグメント数になり，空きページ再利用モードに切り替わってサーチしても，空き領域がない場合です。このようなとき，空回り回数がカウントアップされます。空き領域がないのにサーチしているため，むだにサーチ処理をしていることになります。<br>また，空き領域の再利用のページサーチ空回り回数とページサーチモードの切り替え回数が共に増加している場合は，全く空き領域がない状態で空き領域の再利用が実行されていることになります。 | 指定するセグメント数やセグメントサイズの見直し，又は空き領域の再利用機能の使用中止を検討してください。 |

| 項目 | 説明 | 対処 |
|---|---|---|
| 使用中セグメント数 | 表が使用しているセグメントに空き領域がなくなると未使用セグメントが確保されるため，表の使用中セグメント数が増加します。空き領域の再利用のページサーチ空回り回数と同じ数だけ増加している場合，空き領域がないのに再利用しようとしてむだにサーチ処理をしていることになります。 | － |

（凡例）－：該当しません。

## 14.5.7 注意事項

- 次の場合，空き領域の再利用機能は動作しません。
  - ハッシュ分割表のリバランス機能でデータを格納するとき
  - データディクショナリ表を格納するとき
  - データロードやデータベース再編成ユティリティ（pdrorg）で表にデータを格納するとき
  - ユーザ LOB 用 RD エリアのとき
- 空き領域の再利用機能使用時，削除による空き領域が連続しない場合，連続している場合と比較するとページのサーチ処理は遅くなります。この場合，空き領域の再利用機能使用の中止や，データベース再編成ユティリティ（pdrorg）によるデータの再編成を検討してください。
- 非 FIX 表の場合，空き領域再利用機能を適用していても，次のような更新を行うとデータの追加と削除が同量でもセグメント数が増加する場合があります。
  - **↓（ページ長－ 48）÷（行長＋ 2）↓**件のデータを挿入し，行長が短くなるデータ（NULL 値含む），又は別のページに分岐するようなデータに更新した後削除する操作を繰り返す。

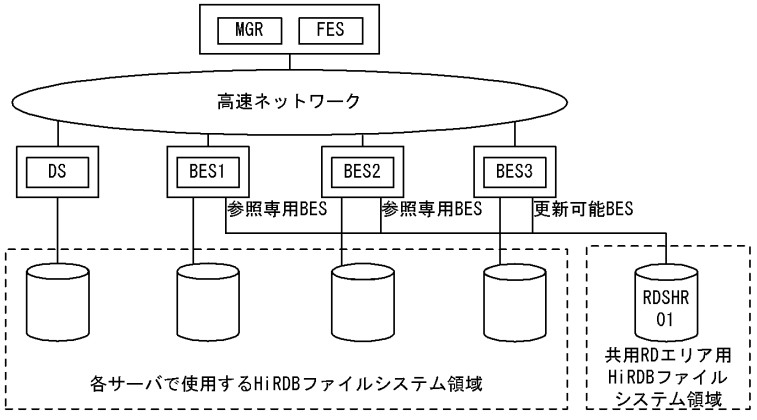
  そのため，定期的にデータベース再編成ユティリティ（pdrorg）によるデータの再編成を実施してください。
- 空きページ再利用モードでサーチ実行時でも，同一トランザクションで削除された領域は再利用されません。

# 14.6 共用 RD エリア（HiRDB/ パラレルサーバ限定）

通常，バックエンドサーバがアクセスできるのは，バックエンドサーバ下の RD エリアだけです。そのため，表の検索や更新時はできるだけ表を分割して格納した方が並列処理ができ，処理速度が向上します。しかし，複数のトランザクションからアクセスが集中し，かつ分割が難しい表などの場合，共用 RD エリアに格納することで並列処理の効率が上がります。**共用 RD エリア**とは，すべてのバックエンドサーバからアクセスできるユーザ用 RD エリアのことです。共用 RD エリアに格納する表を**共用表**，インデクスを**共用インデクス**といい，すべてのバックエンドサーバから参照できます。共用 RD エリアに格納できるのは，共用表及び共用インデクスだけです。共用 RD エリアの概要を次の図に示します。

なお，共用 RD エリアを定義できるのは HiRDB/ パラレルサーバだけです。

**図 14-5 共用 RD エリアの概要**




〔説明〕

共用 RD エリア RDSHR01 は，BES1 ～ 3 すべてのバックエンドサーバから参照できます。ただし，共用表の更新ができるのは更新可能バックエンドサーバ（BES3）だけです。BES1 及び 2 は参照専用バックエンドサーバです。

## (1) 効果

全バックエンドサーバが共用 RD エリアにアクセスできるため，並列処理の効率が上がります。

## (2) 適用基準

次の場合に共用 RD エリアの使用をお勧めします。

- 複数のトランザクションからアクセスが集中するが，分割が難しい表の場合
- 結合処理のような複雑な検索処理をする場合

## (3) 定義方法

共用 RD エリアを使用するには，次のように指定します。

- pd_shared_rdarea_use オペランドに Y を指定します。
- pdfmkfs コマンドの -k オプション（使用目的）に SDB を指定します。また，全バックエンドサーバから同じパス名でアクセスできるようにアクセスパスを設定します。
- データベース初期設定ユティリティ（pdinit），又はデータベース構成変更ユティリティ（pdmod）の

create rdarea 文に shared を指定し，ユーザ用 RD エリアの定義をします。また，更新可能バックエンドサーバを server name オペランドに指定します。server name オペランドに指定しなかったバックエンドサーバはすべて参照専用バックエンドサーバになります。

**定義時の注意**

- 共用 RD エリアは，pd_max_rdarea_no オペランドで指定した RD エリアの最大数まで定義できます。ただし，共用 RD エリアは全バックエンドサーバの RD エリアの数に加算されます。
- 異なるバックエンドサーバが更新可能バックエンドサーバである共用 RD エリアを，同一 HiRDB ファイルシステム領域に定義してはいけません。
- 共用 RD エリアは，共用 RD エリア用の HiRDB ファイルシステム領域に定義します。pdfmkfs -k コマンドに SDB を指定してください。なお，共用 RD エリア用の HiRDB ファイルシステム領域には共用 RD エリア以外は定義できません。

図 14-5 の場合の，データベース構成変更ユティリティ（pdmod）の制御文の例を次に示します。

```
create shared rdarea RDSHR01 globalbuffer buf01 for user used by PUBLIC
  server name BES3   …更新可能バックエンドサーバの指定
  open attribute INITIAL
  page 4096 characters
  storage control segment 20 pages
  file name "¥HiRDB¥DATABASE¥SHR1¥rdshr01_f01"   …ファイル名
          initial 10000 segments ;
```

## (4) 共用 RD エリアの更新

共用 RD エリアを更新する場合，LOCK 文で IN EXCLUSIVE MODE を指定し，全バックエンドサーバの共用 RD エリアに排他を掛けなければ実行できません。ただし，インデクスキー値を変更しない UPDATE 文は，LOCK 文を発行しないで実行できます。共用表の更新については，「12.18.3　共用表の操作」を参照してください。なお，共用表及び共用インデクスの更新は COMMIT 文発行時にディスクに書き込まれます。

## (5) 共用 RD エリアの閉塞状態の管理

共用 RD エリアへのアクセスは，各バックエンドサーバで個別に管理されます。このため，障害発生時に，バックエンドサーバ間で閉塞状態が異なる場合があります。データベース構成変更ユティリティ（pdmod）や，データベース回復ユティリティ（pdrstr）を実行する場合は，pdhold コマンドで全バックエンドサーバ下の共用 RD エリアの閉塞状態を一致させる必要があります。なお，pddbls -m **コマンド**で全バックエンドサーバの共用 RD エリアの状態を表示できます。

## (6) 共用 RD エリアに対するユティリティ及び運用コマンドの実行

ユティリティ及び運用コマンドで共用 RD エリア，共用表，又は共用インデクスを対象とする場合，HiRDB が内部的に LOCK TABLE 文を発行し，全バックエンドサーバ下の共用 RD エリアに排他を掛けることがあります。このため，共用 RD エリア内の表やインデクスにアクセス中の業務があると，デッドロック，又はサーバ間のグローバルデッドロックが発生することがあります。ユティリティ及び運用コマンド実行時は，対象となる共用 RD エリアをコマンド閉塞しておいてください。

## (7) 共用 RD エリア使用上の制限事項

1. 系切り替え機能を使用する場合，更新可能バックエンドサーバがあるユニットは次のように配置してください。
   - 参照専用バックエンドサーバと異なるホストに配置
   - 系を切り替えたときに同一ホスト内で参照バックエンドサーバと混在しないように切り替え先を配置

参照専用バックエンドサーバは，共用 RD エリアのディスクボリュームをクラスタソフトウェアの管理リソースにしないようにしてください。

2. 共用 RD エリアは全バックエンドサーバに配置されるため，フロータブルサーバは設置できません。
3. レプリケーションの反映先に共用表は指定できません。
4. ローカルバッファで共用 RD エリアの表やインデクスを更新する場合は，LOCK TABLE 文を発行して更新してください。LOCK TABLE 文を発行しないで更新していると，サーバプロセスが異常終了してトランザクション回復プロセスが回復処理するとき，グローバルバッファで回復対象になる更新ページを保持できないことがあります。更新ページを保持できないと回復できないので，アボートコード Phb3008 を出力してユニットは異常終了します。この場合，HiRDB を再開始してください。

# 15 HiRDB のメモリ所要量

この章では，HiRDB/ シングルサーバ及び HiRDB/ パラレルサーバのメモリ所要量の求め方について説明します。

15.1 HiRDB/ **シングルサーバのメモリ所要量の見積もり**

15.2 HiRDB/ **パラレルサーバのメモリ所要量の見積もり**

# 15.1 HiRDB/ シングルサーバのメモリ所要量の見積もり

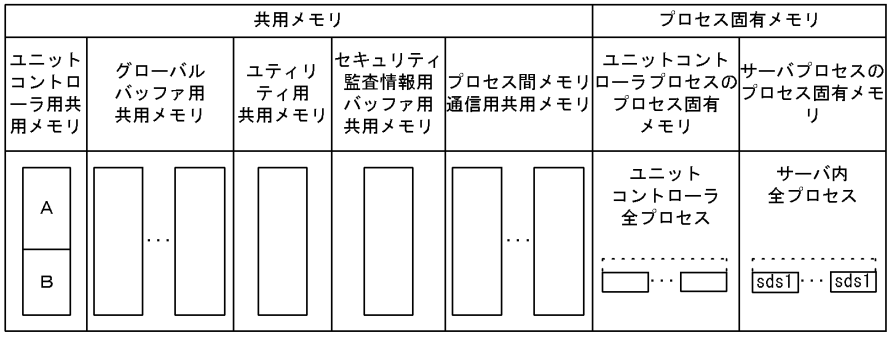
ここでは，HiRDB/ シングルサーバのメモリ所要量の見積もり方法について説明します。ここで説明する項目を次に示します。

- メモリ配置
- メモリ所要量の計算式
- ユニットコントローラが使用する共用メモリの計算式
- シングルサーバが使用する共用メモリの計算式
- グローバルバッファが使用する共用メモリの計算式
- SQL 実行時に必要なメモリ所要量の計算式
- SQL 前処理時に必要なメモリ所要量の計算式
- BLOB 型データの検索又は更新時に必要なメモリ所要量の計算式
- ブロック転送又は配列 FETCH で必要なメモリ所要量の計算式

## 15.1.1 メモリ配置

HiRDB/ シングルサーバのメモリ配置を次の図に示します。

**図 15-1 HiRDB/ シングルサーバのメモリ配置**




（凡例）A：ユニットコントローラの各プロセス使用分
B：シングルサーバのプロセス使用分
sds：シングルサーバ

HiRDB/ シングルサーバの共用メモリの詳細を次の表に示します。

**表 15-1 HiRDB/ シングルサーバの共用メモリの詳細**

| 項目 | 共用メモリの種類 | | | | |
|---|---|---|---|---|---|
| | ユニットコントローラ用共用メモリ | グローバルバッファ用共用メモリ | ユティリティ用共用メモリ | セキュリティ監査情報用バッファ用共用メモリ | プロセス間メモリ通信用共用メモリ |
| 使用目的 | システム制御 | グローバルバッファ | ユニットコントローラとユティリティとの通信 | セキュリティ監査情報用バッファ | クライアント－サーバプロセス間メモリ通信 |

| 項目 | 共用メモリの種類 | | | | |
|---|---|---|---|---|---|
| | ユニットコントローラ用共用メモリ | グローバルバッファ用共用メモリ | ユティリティ用共用メモリ | セキュリティ監査情報用バッファ用共用メモリ | プロセス間メモリ通信用共用メモリ |
| 使用プロセス | 全 HiRDB プロセス | シングルサーバ | ユティリティプロセス | シングルサーバ | シングルサーバ，クライアントプロセス |
| セグメント数 | 1 個 | • グローバルバッファの動的変更機能を使用しない場合<br>1 ～ 16 個<br>• グローバルバッファの動的変更機能を使用している場合<br>32 ビットモード：1 ～ 516 個<br>64 ビットモード：1 ～ 1,016 個 | 1 個 | 1 個 | 環境変数 PDIPC=MEMORY で接続中のクライアント数（0 ～ 2000）× 2 個 |
| 1 セグメントの上限 | 「表 15-2 HiRDB/ シングルサーバが使用するメモリ所要量」を参照してください。ただし，OS の環境によっては確保できない場合があります。 | SHMMAX オペランドの値でセグメントを分割します。ただし，OS の環境によっては確保できない場合があります。 | 「表 15-2 HiRDB/ シングルサーバが使用するメモリ所要量」を参照してください。 | 「表 15-2 HiRDB/ シングルサーバが使用するメモリ所要量」を参照してください。ただし，OS の環境によっては確保できない場合があります。 | 「表 15-2 HiRDB/ シングルサーバが使用するメモリ所要量」を参照してください。ただし，OS の環境によっては確保できない場合があります。 |
| 確保条件 | なし | グローバルバッファ定義が存在すること | pd_utl_exec_mode=1 指定 | pd_aud_file_name オペランドによる監査証跡ファイル用の HiRDB ファイルシステム領域名の指定 | クライアント環境変数 PDIPC=MEMORY で接続中クライアントあり |
| 作成契機 | ユニット起動時（ユーザサーバホットスタンバイまたは高速系切り替え機能使用時の待機ユニットの起動を含む） | • サーバ起動時（高速系切り替え機能使用時の待機ユニットの起動を含む）<br>• pdbufmod -k {add\|upd} 実行時 | ユティリティ実行時 | シングルサーバ起動時 | クライアントとサーバの接続時 |

| 項目 | 共用メモリの種類 | | | | |
|---|---|---|---|---|---|
| | ユニットコントローラ用共用メモリ | グローバルバッファ用共用メモリ | ユティリティ用共用メモリ | セキュリティ監査情報用バッファ用共用メモリ | プロセス間メモリ通信用共用メモリ |
| 削除契機 | ユニット起動時（ユーザサーバホットスタンバイまたは高速系切り替え機能使用時の待機ユニットの起動を含む） | • pdbufmod -k del 実行時<br>• 正常終了又は計画停止の場合：サーバ終了時<br>• 強制停止，異常終了，又は高速系切り替え機能使用時の待機ユニットの停止の場合：次回ユニット起動時 | ユティリティ終了後 10 分後 | シングルサーバ終了時 | クライアントとサーバの接続切り離し時 |
| pdls -d mem による表示 | 表示される | 表示される | 表示される | 表示される | 表示されない |
| pdls -d mem の SHM-OWNER | MANAGER | サーバ名 | UTILITY | AUDDEF | 表示なし |
| 関連オペランド | • pd_sds_shmpool_size | • pd_dbbuff_modify<br>• pdbuffer<br>• SHMMAX | • pd_utl_exec_mode | • セキュリティ監査機能に関するオペランド※ | • PDIPC<br>• PDSENDMEMSIZE<br>• PDRECVMEMSIZE |
| 備考 | — | — | pd_utl_exec_mode=1 の場合だけ作成されます（pd_utl_exec_mode=0 の場合，該当する領域はユニットコントローラ用共用メモリ内に確保されます）。 | — | — |

（凡例）—：該当しません。

注※

詳細はマニュアル「HiRDB Version 8 システム定義」を参照してください。

## 15.1.2 メモリ所要量の計算式

HiRDB/ シングルサーバが使用するメモリ所要量は，次の表に示すすべての項目を加算した値です。

**表 15-2　HiRDB/ シングルサーバが使用するメモリ所要量**

| 項目 | | | メモリ所要量（単位：キロバイト） |
|---|---|---|---|
| プロセス固有領域 | ユニットコントローラ全プロセスが使用するプロセス固有領域 | | **● 32 ビットモードの場合**<br>E ＋↑ ｛(64 ＋ 48 ×（u ＋ 1)）×（pd_max_server_process の値－ b － 3）＋ w｝÷ 1024 ↑<br>**● 64 ビットモードの場合**<br>E ＋↑ ｛(64 ＋ 64 ×（u ＋ 1)）×（pd_max_server_process の値－ b － 3）＋ w｝÷ 1024 ↑<br><br>**●プラグインを使用する場合，加算します。**<br>＋ 1400<br>**●非同期 READ 機能を使用する場合，加算します。**<br>＋ r<br>**● pd_process_terminator オペランドに fixed を指定した場合，加算します。**<br>＋ F ×（pd_process_terminator_max の値－ 1）<br>**●インメモリデータ処理を行う場合，加算します。**<br>＋↑ ｛K ×（pd_max_users の値× 2 + 7)｝÷ 1024 ↑<br>**●通信トレース格納最大数を変更する場合，加算します。**<br>+ ↑ M ÷ 1024 ↑ |
| | シングルサーバプロセスが使用するプロセス固有領域※1 | pd_work_buff_mode= each 指定時 | **● 32 ビットモードの場合**<br>｛G ＋ g ＋（a ＋ 9）× c ＋ h ＋ i ＋ m ＋ p ＋ q ＋ s｝×（b ＋ 3）＋↑（64 ＋ 48 ×（b ＋ 1））÷ 1024 ↑×（v ＋ 3）＋ J<br>**● 64 ビットモードの場合**<br>｛G ＋ g ＋（a ＋ 9）× c ＋ h ＋ i ＋ m ＋ p ＋ q ＋ s｝×（b ＋ 3）＋↑（64 ＋ 64 ×（b ＋ 1））÷ 1024 ↑×（v ＋ 3）＋ J<br>**●インメモリデータ処理を行う場合，加算します。**<br>＋↑ ｛K ×（pd_max_users の値＋ 3)｝÷ 1024 ↑<br>**●通信トレース格納最大数を変更する場合，加算します。**<br>＋↑ P ÷ 1024 ↑ |
| | | pd_work_buff_mode= pool 指定又は省略時 | **● 32 ビットモードの場合**<br>(G ＋ g ＋ a ＋ 9 ＋↑ a ÷ 128 × 0.1 ↑＋ 11 ＋ h ＋ i ＋ m ＋ p ＋ q ＋ s）×（b ＋ 3）＋ n ＋↑（64 ＋ 48 ×（b ＋ 1））÷ 1024 ↑×（v ＋ 3）＋ J<br>**● 64 ビットモードの場合**<br>(G ＋ g ＋ a ＋ 9 ＋↑ a ÷ 128 × 0.1 ↑＋ 15 ＋ h ＋ i ＋ p ＋ q ＋ s）×（b ＋ 3）＋ n ＋↑（64 ＋ 64 ×（b ＋ 1））÷ 1024 ↑×（v ＋ 3）＋ J<br>**●インメモリデータ処理を行う場合，加算します。**<br>＋↑ ｛K ×（pd_max_users の値＋ 3)｝÷ 1024 ↑<br>**●通信トレース格納最大数を変更する場合，加算します。**<br>＋↑ P ÷ 1024 ↑ |
| 共用メモリ | ユニットコントローラ用共用メモリ中，ユニットコントローラが使用する領域 | | ↑ d ÷ 1024 ↑ |
| | ユニットコントローラ用共用メモリ中，シングルサーバが使用する領域 | | e |
| | グローバルバッファ用共用メモリ | | f |
| | インメモリデータ処理用共用メモリ | | L |
| | ユティリティ用共用メモリ | | t |

| 項目 | | メモリ所要量（単位：キロバイト） |
|---|---|---|
| | セキュリティ監査情報用バッファ用共用メモリ | **●システムによる自動計算の場合**<br>↑ 0.3 ＋ MAX｛(H ＋ 100)，(H × 1.2)｝× 0.25 ↑<br>**●ユーザ指定値（pd_audit_def_buffer_size オペランドを指定）にする場合**<br>pd_audit_def_buffer_size の指定値 |
| | プロセス間メモリ通信用共用メモリ※2 | j × k |

注※1

プラグインを使用する場合は，1 シングルサーバプロセス当たり 300 を加算してください。

注※2

クライアント環境定義で PDIPC=MEMORY を指定した場合に加算します。プロセス間メモリ通信機能及びクライアント環境定義については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

なお，HiRDB サーバ又は HiRDB クライアントのどちらかが 32 ビットモードの場合，プロセス間メモリ通信機能で使用する共用メモリは 32 ビット空間内に確保されます。

a：pd_work_buff_size オペランドの値

b：pd_max_users オペランドの値

c：最大作業表数

SQL 文ごとの作業表数を「表 15-3　SQL 文ごとの作業表数の求め方」から求めます。表 15-3 から求めた作業表数のうちで最大のものを最大作業表数とします。

d：「15.1.3　ユニットコントローラが使用する共用メモリの計算式」で求めた値

e：「15.1.4　シングルサーバが使用する共用メモリの計算式」で求めた値

f：「15.1.5　グローバルバッファが使用する共用メモリの計算式」で求めた値

g：SQL 実行時に必要なメモリ所要量

計算式については，「15.1.6　SQL 実行時に必要なメモリ所要量の計算式」を参照してください。

h：SQL 前処理時に必要なメモリ所要量

計算式については，「15.1.7　SQL 前処理時に必要なメモリ所要量の計算式」を参照してください。

i：LOB バッファ一括入出力ワークメモリ

グローバルバッファ定義に LOB 用グローバルバッファを指定している場合だけ（システム共通定義の pdbuffer オペランドに -b を指定している場合），62 キロバイトを加算してください。

j：プロセス間メモリ通信機能を使用するクライアントの最大同時実行数

分からない場合は，プロセス間メモリ通信機能を使用する全クライアント数，又は pd_max_users オペランドの値を代入してください。

k：プロセス間メモリ通信機能を使用する全クライアントのデータ送受信用メモリサイズ（クライアント環境定義の PDSENDMEMSIZE の値＋ PDRECVMEMSIZE の値）の平均値

m：Java 仮想マシンが使用するメモリ所要量

Java ストアドプロシジャ又は Java ストアドファンクションを使用する場合に，Java 仮想マシンが使用するメモリ所要量を加算します。Java 仮想マシンが使用するメモリ所要量は，Java 仮想マシンのオプションや Java 仮想マシンのバージョンによって異なります。Java 仮想マシンが使用するメモ

リ所要量については，Java 仮想マシンのマニュアルを参照してください。

n：作業表用増分メモリサイズ

pd_work_buff_expand_limit オペランドを指定する場合に作業表用増分メモリサイズを加算します。作業表用増分メモリサイズは次に示す計算式から求めます。

**作業表用増分メモリサイズ（キロバイト）＝作業表用増分バッファサイズ＋↑（作業表用の増分バッファサイズ÷ 128）× 0.1 ↑**

- 作業表用増分バッファサイズ（キロバイト）＝ MAX（0，ハッシュジョイン，副問合せのハッシュ実行による作業表用増分バッファサイズ）＋ MAX（0，作業表数の増加による作業表用増分バッファサイズ）
- ハッシュジョイン，副問合せのハッシュ実行による作業表用増分バッファサイズ＝ MIN｛(ハッシュジョイン，副問合せのハッシュ実行をするときの作業表用バッファサイズ－ pd_work_buff_size オペランドの値)，(pd_work_buff_expand_limit オペランドの値－ pd_work_buff_size オペランドの値)｝×ハッシュジョイン，副問合せのハッシュ実行をする同時実行ユーザ数

  ハッシュジョイン，副問合せのハッシュ実行をするときの作業表用バッファサイズの求め方については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。
- 作業表数の増加による作業表用増分バッファサイズ＝ MIN｛(使用作業表数× 128 － pd_work_buff_size オペランドの値)，(pd_work_buff_expand_limit オペランドの値－ pd_work_buff_size オペランドの値)｝×（使用作業表数が pd_work_buff_size オペランドの値÷ 128 以上になるユーザ数）

  使用作業表数＝ MAX（1SQL 文が使用する作業表用ファイルの数，ASSIGN LIST 文が使用する作業表用ファイルの数）

  1SQL 文が使用する作業表用ファイルの数，及び ASSIGN LIST 文が使用する作業表用ファイルの数の求め方については，「18.3　最大ファイル数の見積もり（pdfmkfs -l コマンド）」を参照してください。

p：BLOB 型データ用に必要なメモリ所要量

計算式については，「15.1.8　BLOB 型データの検索又は更新時に必要なメモリ所要量の計算式（HiRDB/ シングルサーバの場合）」を参照してください。

q：サーバ側でブロック転送又は配列 FETCH で必要なメモリ所要量

計算式については，「15.1.9　ブロック転送又は配列 FETCH で必要なメモリ所要量の計算式」を参照してください。

r：非同期 READ 用メモリサイズ

非同期 READ 機能を使用する場合に加算します。次に示す計算式から求めます（単位：キロバイト）。

$$(90 + \sum_{i=1}^{90} \text{RD エリア用 HiRDB ファイルシステム領域管理用メモリ}) \times \text{pd_max_ard_process の値}$$

RD エリア用 HiRDB ファイルシステム領域管理用メモリは計算値の大きい順に 90 領域を計算に使用します。サーバで使用する領域数が 90 領域に満たない場合，その領域まで計算します。

HiRDB ファイルシステム領域管理用メモリはそれぞれの領域の初期設定時のパラメタを使用して，次の計算式から求めます（単位：キロバイト）。

$$\{(\text{ファイル数}^{※1} + \text{増分数}^{※2}) \div 64\} \times 1.5^{※3}$$

注※ 1　pdfmkfs -l 指定値です。
注※ 2　pdfmkfs -e 指定値です。
注※ 3　領域サイズ（pdfmkfs -n 指定値）が 2048 以上の場合に乗算します。

s：HiRDB ファイルシステム用メモリサイズ
次に示す計算式から求めます（単位：キロバイト）。

```
347 ＋作業表用 HiRDB ファイルシステム領域管理用メモリ＋
システムログ用 HiRDB ファイルシステム領域管理用メモリ＋
90
ΣRD エリア用 HiRDB ファイルシステム領域管理用メモリ
i=1
```

作業表用及びシステムログ用 HiRDB ファイルシステム領域管理用メモリは，サーバで使用する HiRDB ファイルシステム領域で計算値が最大になるものを使用します。RD エリアの場合は計算値の大きい順に 90 領域を計算に使用します。サーバで使用する領域数が 90 領域に満たない場合，その領域まで計算します。
HiRDB ファイルシステム領域管理用メモリはそれぞれの領域の初期設定時のパラメタを使用して，次の計算式から求めます（単位：キロバイト）。

$\{(\text{ファイル数}^{※1} + \text{増分数}^{※2}) \div 64\} \times 1.5^{※3}$

注※ 1　pdfmkfs -l 指定値です。
注※ 2　pdfmkfs -e 指定値です。
注※ 3　領域サイズ（pdfmkfs -n 指定値）が 2048 以上の場合に乗算します。

t：pd_utl_exec_mode の値が 0 の場合：0
pd_utl_exec_mode の値が 1 の場合：↑｛(b × 2000 ＋ 136）÷ 1024｝↑× 1024

u：ユニット制御情報定義として有効な pd_module_trace_max の値

v：シングルサーバ定義として有効な pd_module_trace_max の値

w：HiRDB の再開始用メモリサイズ
このメモリサイズを確保できないと，HiRDB の再開始に失敗します。次の計算式から求めます（単位：バイト）。

```
A ＋ B

● pd_dbsync_point オペランドに commit を指定している場合，加算します。
＋ 112 ×（pd_max_users の値× 2 ＋ 7）
● raw I/O 機能を使用した HiRDB ファイルシステム領域に作成された RD エリアを格納した HiRDB ファイルシステム領域数が 1001 以上の場合に加算します。
＋ D
```

HiRDB の再開始用メモリサイズを求める計算に使用する変数を次に示します。

| 変数 | 値 |
|---|---|
| A | ● 32 ビットモードの場合<br>246762 ＋ 4 × pd_max_rdarea_no の値<br>＋｛48 ×（pd_max_rdarea_no の値＋表数）＋ 304｝×（pd_max_users の値× 2 ＋ 7）<br>● 64 ビットモードの場合<br>305274 ＋ 8 × pd_max_rdarea_no の値<br>＋｛64 ×（pd_max_rdarea_no の値＋表数）＋ 512｝×（pd_max_users の値× 2 ＋ 7）<br><br>**表数：**62 ＋ MAX｛pd_max_access_tables の値，500｝ |
| B | b1 × X ＋ b2 × Y<br><br>b1：サーバ用ステータスファイルのレコード長＜ 4096 の場合<br>MAX（(↓（3400 ÷（(↓（(レコード長－ 40）－ 308）÷ 20 ↓）<br>＋（↓（レコード長－ 40）÷ 20 ↓）×（MAX（↓ 4096 ÷レコード長↓，2）－ 1））<br>＋ 0.7）↓），1）× MAX( ↓ 4096 ÷レコード長↓，2）×（レコード長－ 40）<br>4096 ≦サーバ用ステータスファイルのレコード長＜ 12288 の場合<br>MAX（↓（3400 ÷（↓（(((レコード長－ 40）－ 308）÷ 20）↓）＋ 0.7）↓，1）<br>×（レコード長－ 40）<br>12288 ≦サーバ用ステータスファイルのレコード長の場合<br>MAX（↓（3400 ÷（↓（(((レコード長－ 40）－ 836）÷ 20）↓）＋ 0.7）↓，1）<br>×（レコード長－ 40）<br>X：RD エリア数≦ 3400 の場合：1<br>3401 ≦ RD エリア数≦ 6800 の場合：2<br>6801 ≦ RD エリア数の場合：3<br>b2：サーバ用ステータスファイルのレコード長＜ 4096 の場合<br>（↓（5662310 ÷（(↓（(レコード長－ 40）－ 308）÷ 20 ↓）<br>＋（↓（レコード長－ 40）÷ 20 ↓）×（MAX（↓ 4096 ÷レコード長↓，2）－ 1））<br>＋ 0.7）↓）× MAX（↓ 4096 ÷レコード長↓，2）×（レコード長－ 40）<br>4096 ≦サーバ用ステータスファイルのレコード長＜ 12288 の場合<br>↓（5662310 ÷（↓（(((レコード長－ 40）－ 308）÷ 20）↓）＋ 0.7）↓<br>×（レコード長－ 40）<br>12288 ≦サーバ用ステータスファイルのレコード長の場合<br>↓（5662310 ÷（↓（(((レコード長－ 40）－ 836）÷ 20）↓）＋ 0.7）↓<br>×（レコード長－ 40）<br>Y：RD エリア数≦ 10200 の場合：0<br>10201 ≦ RD エリア数≦ 5672510 の場合：1<br>5672511 ≦ RD エリア数≦ 11334820 の場合：2<br>11334821 ≦ RD エリア数の場合：3 |
| D | ● 32 ビットモードの場合<br>12012 ×（↑（raw I/O 機能を使用した HiRDB ファイルシステム領域に作成された RD エリアを格納した HiRDB ファイルシステム領域数－ 1000）÷ 1000 ↑）<br>● 64 ビットモードの場合<br>16016 ×（↑（raw I/O 機能を使用した HiRDB ファイルシステム領域に作成された RD エリアを格納した HiRDB ファイルシステム領域数－ 1000）÷ 1000 ↑） |

E，F，G：固定値

この値は OS によって異なります。OS ごとの値を次に示します（単位：キロバイト）。

| OS | E の値 | F の値 | G の値 |
|---|---|---|---|
| Windows（32 ビットモード） | 74,000 | 3,300 | 5,200 |
| Windows Server 2003 (IPF) | 167,000 | 5,400 | 11,700 |
| Windows (x64) | 104,900 | 4,700 | 11,300 |

H：セキュリティ監査機能の監査証跡の絞り込み条件で指定したオブジェクトの数

Ｊ：シンクポイント出力同期制御情報取得機能使用時に必要なメモリ（単位：バイト）
pd_dbbuff_trace_level オペランドに 1 を指定し，かつ pd_dfw_awt_process オペランドの指定を省略している場合に，次の値を加算します。

32 ビットモードの場合：
320 ×シングルサーバに定義したグローバルバッファの数

64 ビットモードの場合：
640 ×シングルサーバに定義したグローバルバッファの数

K：pd_max_resident_rdarea_no オペランドに 1 以上を指定している場合に，次の値を加算します。
1648 ＋ 16 × pd_max_resident_rdarea_no の値＋ 16 × pd_max_resident_rdarea_shm_no の値

Ｌ：インメモリデータ処理で必要なメモリ所要量
計算式については，「15.1.10　インメモリデータ処理で必要なメモリ所要量」を参照してください。

M：通信トレース処理で必要なメモリ所要量

32 ビットモードの場合：
（16 ×（N － 1024）× 2）×（pd_max_server_process 値－ pd_max_users 値－ 3）

64 ビットモードの場合：
（32 ×（N － 1024）× 2）×（pd_max_server_process 値－ pd_max_users 値－ 3）

N：ユニット制御情報定義として有効な pd_pth_trace_max 値です。オペランド指定値を 2 のべき乗の値に切り上げた値になります。

P：通信トレース処理で必要なメモリ所要量

32 ビットモードの場合：
（16 ×（Q － 1024）× 2）×（pd_max_users 値＋ 3）

64 ビットモードの場合：
（32 ×（Q － 1024）× 2）×（pd_max_users 値＋ 3）

Q：シングルサーバ定義として有効な pd_pth_trace_max 値です。オペランド指定値を 2 のべき乗の値に切り上げた値になります。

表 15-3 SQL 文ごとの作業表数の求め方

| SQL 文 | 作業表数の求め方 |
|---|---|
| SELECT 文<br>INSERT 文（問合せ指定の場合） | 1. ～ 8. の指定がない場合：0<br>1. ～ 8. の指定がある場合：該当する作業表数をすべて加算した値<br>1. **複数の表を結合して検索する場合**<br>増加作業表数＝（結合表数－ 1）× 2 ＋ 1<br>2. ORDER BY **句を指定する場合**<br>増加作業表数＝ 2<br>3. GROUP BY **句を指定する場合**<br>増加作業表数＝ GROUP BY 句指定数<br>4. DISTINCT **句を指定する場合**<br>増加作業表数＝ DISTINCT 句指定数<br>5. UNION **句，**UNION ALL **句又は** EXCEPT[ALL] **句を指定する場合**<br>増加作業表数＝（UNION 又は UNION ALL 句指定数）× 2 ＋ 1<br>6. **探索条件中にインデクスを定義した列がある場合**<br>増加作業表数＝探索条件中のインデクスを定義した列数<br>7. FOR UPDATE **句又は** FOR READ ONLY **句を指定する場合**<br>増加作業表数＝ 1<br>8. **副問合せ（限定述語）を指定する場合**<br>増加作業表数＝副問合せ指定数 |
| UPDATE 文<br>DELETE 文 | 探索条件中のインデクスを定義した列数＋ 1 |
| DROP SCHEMA 文<br>DROP TABLE 文<br>DROP INDEX 文<br>CREATE INDEX 文<br>REVOKE 文でアクセス権限を取り消す場合 | 2 |

## 15.1.3 ユニットコントローラが使用する共用メモリの計算式

### (1) 32 ビットモードの HiRDB の場合

HiRDB/ シングルサーバの開始から終了までの間にユニットコントローラが使用する共用メモリは，次に示す HiRDB のプロセスの項目すべてを加算した値です。

なお，ユニットコントローラ全体の共用メモリサイズは 2 ギガバイト以内になるようにしてください。

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| スケジューラ | pd_utl_exec_mode **の値が** 0 **の場合**<br>｛↑（432 ＋ 304 × n）÷ 1024 ↑＋ 352 ＋ x｝× 1024<br>pd_utl_exec_mode **の値が** 1 **の場合**<br>｛↑（432 ＋ 304 × n）÷ 1024 ↑＋↑（m × 2000 ＋ 136）÷ 1024 ↑＋ y｝× 1024<br>x：シングルサーバの場合：116 ＋ 5 ×（m ＋ 3）＋ 14<br>y：シングルサーバの場合：5 ×（m ＋ 3）＋ 14<br>m：pd_max_users の値<br>n：ユニット内のサーバ数＋ユニット内ユティリティサーバ数＋ 1<br>ユニット内ユティリティサーバ数：28 ＋ 12 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| ロックサーバ | （192 ＋ 48 ＋ c ＋ d ＋ 48 ＋ 4096 ＋ g ＋ 48 ＋ i ＋ 48 ＋ 4096 ＋ 48 ＋ n ＋ 16 ＋ t ＋ u ＋ 16）<br>× pd_lck_pool_partition の値<br>c：pd_lck_hash_entry を省略，又は 0 を指定している場合：<br>（↓（8 ＋ 4 × MAX（((p ＋ 3）×（pd_max_access_tables の値＋ 4 ＋ 5 × 2）<br>＋↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 6）÷ 10，11261））÷ 16 ↓<br>＋ 1）× 16<br>pd_lck_hash_entry に 2 以上の素数でない値を指定している場合：<br>（↓（8 ＋ 4 × pd_lck_hash_entry の値を超えない最大の素数）÷ 16 ↓＋ 1）<br>× 16<br>pd_lck_hash_entry に 1，又は素数を指定している場合：<br>（↓（8 ＋ 4 × pd_lck_hash_entry の値）÷ 16 ↓＋ 1）× 16<br>d：((ｐ ＋ 3）×（pd_max_access_tables の値＋ 4 ＋ 5 × 2）＋<br>↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 6）× 96<br>g：pd_utl_exec_mode の値 =1，かつ p ＞ 32 の場合：<br>((p ＋ 3）× 3 ＋ p）× 256<br>pd_utl_exec_mode の値 =0，又は p ≦ 32 の場合：<br>((p ＋ 3）× 3 ＋ 32）× 256<br><br>i：pd_utl_exec_mode の値 =1，かつ p ＞ 32 の場合：<br>((↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 8 ＋（(ｐ ＋ 3）<br>×（pd_max_access_tables の値＋ 4））× 2）<br>＋ p ×（pd_max_rdarea_no の値＋ 1）＋（p ＋ 3）× 2 × 2 × 5）<br>を偶数に切り上げた値× 64<br>pd_utl_exec_mode の値 =0，又は p ≦ 32 の場合：<br>((↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 8 ＋（(p ＋ 3）<br>×（pd_max_access_tables の値＋ 4））× 2）<br>＋ 32 ×（pd_max_rdarea_no の値＋ 1）＋（p ＋ 3）× 2 × 2 × 5）<br>を偶数に切り上げた値× 64<br>n：pd_utl_exec_mode の値 =1，かつ p ＞ 32 の場合：<br>((p ＋ 3）× 3 ＋ p）× 48<br>pd_utl_exec_mode の値 =0，又は p ≦ 32 の場合：<br>((p ＋ 3）× 3 ＋ 32）× 48<br><br>p：pd_max_users の値<br>t：pd_utl_exec_mode の値 =1，かつ ｐ ＞ 32 の場合：<br>32 ＋（(p ＋ 3）× 3 ＋ p）×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 4<br>pd_utl_exec_mode の値 =0，又は p ≦ 32 の場合：<br>32 ＋（(p ＋ 3）× 3 ＋ 32）×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 4<br>u：pd_utl_exec_mode の値 =1，かつ ｐ ＞ 32 の場合：<br>32 ＋（(↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 8<br>＋（(p ＋ 3）×（pd_max_access_tables の値＋ 4））× 2）<br>＋ p ×（pd_max_rdarea_no の値＋ 1）＋（p ＋ 3）× 2 × 2 × 5）<br>を偶数に切り上げた値×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 4<br>pd_utl_exec_mode の値 =0，又は p ≦ 32 の場合：<br>32 ＋（(↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 8 ＋<br>((p ＋ 3）×（pd_max_access_tables の値＋ 4））× 2）<br>＋ 32 ×（pd_max_rdarea_no の値＋ 1）＋（p ＋ 3）× 2 × 2 × 5）<br>を偶数に切り上げた値×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 4 |
| トランザクションサーバ | 288 ＋ 32 ＋ 192 × m × 2 ＋ 1028<br>＋（420 ＋ 564 ＋ 256 ＋ 384 × 2 ＋ 128）×（m × 2 ＋ 7）＋ 256 × 2<br>m：pd_max_users の値 |
| タイマサーバ | 32 ×（pd_max_users の値＋ 3）×（1 ＋ユニット内ユティリティサーバ数＋ 1）＋ 1440<br><br>ユニット内ユティリティサーバ数：26 ＋ 12 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| 統計ログサーバ | 384 ＋ 128 × 16 ＋ 32 ＋ 288 × 2 ＋ 1024 ＋ 128 × 3 ＋ pd_stj_buff_size の値× 1024 × 3 ＋ 64 ＋ 4096 ＋ 8192 |
| プロセスサーバ | 160 ＋ 512 × a ＋ 80 ＋ 256 ＋（pd_max_server_process の値＋ 50）×（256 ＋ 144）＋ 16 ＋ 8 × 16 ＋ 16 ＋ 16 ＋ 48 ＋ 48 ×（pd_module_trace_max の値＋ 1）<br><br>a：シングルサーバの場合：125 |
| システムマネジャ | 640 ＋（44 ＋ 4）× a × 2 ＋（100 ＋ 4）×（b ＋ 30 ＋ 2）＋（100 ＋ 4）×（c ＋ 1）＋ 40 × b × 14 ＋ 256 ＋ 256 ＋ 36 × d ＋ 12 × e ＋ 8 ＋ 5844 ＋ 212 ＋ f ＋ 16 ＋ 1024 ＋ 272 × h<br><br>a：シングルサーバの定義数<br>b：ユニット内のシングルサーバ数<br>c：ユニット数<br>d：pdunit オペランドの -c オプションの指定数<br>e：pdcltgrp オペランド指定数<br>f：2052 ＋ 128 ×（g ＋ 3）<br>g：シングルサーバの場合：92<br>h：pd_security_host_group オペランドに指定したホストに対応する IP アドレスの数<br>pd_security_host_group オペランドを指定していない場合は 0 |
| ネームサーバ | 169984 |
| ノードマネジャ | ↑（1152 ＋ 288 ×全ユニット数＋ 80 ×全サーバ数＋ 1536 ＋ 800<br>＋ 56 ×ユニット内のサーバ数＋ 240 × A ＋ 44 × A ＋ 28 × A ＋ 16 × B ＋ 32）<br>÷ 1024 ↑× 1024<br>A：pd_utl_exec_mode = 0 の場合：1024<br>pd_utl_exec_mode = 1 で，ユニット内にシングルサーバがある場合：pd_max_users の値× 10 ＋ 400<br>pd_utl_exec_mode = 1 で，ユニット内にシングルサーバがない場合：pd_max_users の値× 7<br>なお，A の値が 1024 を超えない場合は，A を 1024 に置き換えて計算してください。<br>B：pdcltgrp オペランドを指定しない場合：0<br>pdcltgrp オペランドを指定する場合：pdcltgrp オペランドの指定数＋ 1 |
| I/O サーバ | ↑（28 ＋（↑（32 ＋ A）÷ 32 ↑× 32））÷ 128 ↑× 128<br>**A：pd_large_file_use=N を指定した場合**<br>3248 ＋（14 ＋ 16）× 808 ＋ 1 × 272 ＋ 534 × 272 ＋ 16 × 272 ＋ pd_max_file_no 値× 808<br>**pd_large_file_use=Y（又は省略）を指定した場合**<br>A ＝ 3248 ＋（14 ＋ 16）× 972 ＋ 1 × 276 ＋ 534 × 276 ＋ 16 × 276 ＋ pd_max_file_no 値 × 972 |
| ログサーバ | 32 ＋ 48 ＋ 128 × 19 ＋ 384 ＋ 128 × 7 ＋ 1024 ＋ 512<br>＋↑（128 ＋ 256 ＋ 160 ＋ 8 ＋ 64）÷ pd_log_rec_leng の値↑× pd_log_rec_leng の値<br>＋ 64 ＋ 4096 × 2 ＋（736 ＋ 512）× B<br>＋ 128 × pd_log_write_buff_count の値<br>＋（pd_log_write_buff_count の値＋ A）<br>×↑｛pd_log_max_data_size の値＋（68 ＋ 44 ＋ 96 ＋ 160)｝÷ 4096 ↑× 4096<br>＋ C ＋↑｛512 ＋ 256 ＋ 128 × B ＋ 464 × B｝÷（8192 － 128）↑× 8192<br><br>A：16<br>B：pdlogadfg -d sys オペランドの指定数<br>C：512 |
| シンクポイントダンプサーバ | ↑（368 ＋ 1456 × 2）÷ 1024 ↑× 1024<br>＋↑｛(96 ＋ 80 ＋ 208 ＋ 208）＋ 192 ×（pdlogadfg -d spd の指定数）<br>＋ 416 ×（pdlogadpf -d spd の指定数）＋ 1023｝<br>÷ 1024 ↑× 1024 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| ユニット共通 | a ＋ b ＋ 64 ＋（m ＋ 3）× c ＋ 64 ＋ 48 ＋ d ＋ e<br>＋（pd_max_server_process の値＋ 100）×（48 ＋ 16）＋ 32<br>＋（pd_max_server_process の値＋ 100 ＋ 384）× 32 ＋ 32 ＋ f ＋ g ＋ h ＋ i<br>＋（pd_max_server_process の値＋ 127）× 32 ＋ 32<br><br>a：25360 ＋ p × 4<br>b：2988<br>c：1952<br>d：32 × 32<br>e：64 ＋ 64 ×｛(m ＋ 3）× 2<br>＋ MAX（5, ↓［m ＋ 3］÷ 10 ↓）＋ 7｝<br>f：512 ×（13 ＋ 3）× 2<br>g：｛(96 ＋ pd_lck_until_disconnect_cnt の値× 112 ＋ 4095）÷ 4096｝<br>× 4096 × 2<br>h：↑（pd_registered_port で指定したポート番号の数× 16 ＋ 32<br>＋ 1023）÷ 1024 ↑× 1024<br>pd_registered_port を指定していない場合は 0<br>i：k が 2 以上の場合は 32 ＋（8 ＋ 8 × k）× n<br>それ以外の場合は 0<br>k：pd_lck_pool_partition の値<br>m：pd_max_users の値<br>n：(m+3）× 2 ＋ MAX｛5, ↓（m ＋ 3）÷ 10 ↓｝＋ 7<br>p：pd_dbbuff_modify に Y を指定している場合：16 ＋ pd_max_add_dbbuff_shm_no の値＋ pd_max_resident_rdarea_shm_no の値<br>上記以外：16 ＋ pd_max_resident_rdarea_shm_no の値 |
| トランザクションログサーバ | 1024 ＋ 512 × A<br>＋｛<br>128 × B ＋ 128<br>＋［F ＋↑（128 ＋ 256 ＋ 8 ＋ 224）÷ pd_log_rec_leng の値↑× pd_log_rec_leng の値<br>＋↑（pd_log_max_data_size の値＋ 68 ＋ 44 ＋ 96 ＋ 160）<br>÷ pd_log_rec_leng の値↑× pd_log_rec_leng の値］× D<br>＋ E ＋（48 ＋ 8）× B × 2<br>｝×ユニット内のサーバ数<br>＋ 584 × B ＋ 128 × B ＋ 64 × B × C ＋ 128<br>＋｛<br>F ＋↑（128 ＋ 256 ＋ 8 ＋ 224）÷ pd_log_rec_leng の値↑× pd_log_rec_leng の値<br>＋↑（pd_log_max_data_size の値＋ 68 ＋ 44 ＋ 96 ＋ 160）<br>÷ pd_log_rec_leng の値↑× pd_log_rec_leng の値<br>｝<br>＋ E ＋（48 ＋ 8）×（B × 2 ＋ 2）<br><br>A：2<br>B：7 ＋ pd_max_users の値× 2<br>C：1<br>D：シングルサーバの場合，pd_log_rollback_buff_count の値が 0 のときは 8,<br>それ以外のときは pd_log_rollback_buff_count の値<br>E：512<br>F：512 |
| ステータスサーバ | ↑ 64 ÷ 32 ↑× 32 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| 監査証跡管理サーバ | ↑ A ÷ 1024 ↑× 1024<br>A：pd_aud_file_name オペランドの指定がない場合は 640<br>pd_aud_file_name オペランドの指定がある場合は 640 ＋（304 × 200）+B ＋ C<br>B：pd_aud_async_buff_size オペランドの値が 0 の場合は 0<br>pd_aud_async_buff_size オペランドの値が 4096 以上の場合は次の計算値<br>（160 × pd_aud_async_buff_count オペランドの値）<br>+ {(↑ pd_aud_async_buff_size オペランドの値÷ 4096 ↑× 4096）<br>× pd_aud_async_buff_count オペランドの値} ＋ 4096<br>C：pd_aud_auto_loading オペランドに N を指定している場合は 0<br>pd_aud_auto_loading オペランドに Y を指定している場合は 256 × 2 ＋ 240 |

## (2) 64 ビットモードの HiRDB の場合

HiRDB/ シングルサーバの開始から終了までの間にユニットコントローラが使用する共用メモリは，次に示す HiRDB のプロセスの項目すべてを加算した値です。

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| スケジューラ | pd_utl_exec_mode の値が 0 の場合<br>{↑（432 ＋ 304 × n）÷ 1024 ↑＋ 352 ＋ x} × 1024<br>pd_utl_exec_mode の値が 1 の場合<br>{↑（432 ＋ 304 × n）÷ 1024 ↑＋↑（m × 2000 ＋ 136）÷ 1024 ↑＋ y} × 1024<br>x：シングルサーバの場合：116 ＋ 5 ×（m ＋ 3）＋ 14<br>y：シングルサーバの場合：5 ×（m ＋ 3）＋ 14<br>m：pd_max_users の値<br>n：ユニット内のサーバ数＋ユニット内ユティリティサーバ数＋ 1<br>ユニット内ユティリティサーバ数：28 ＋ 12 |
| ロックサーバ | (272 ＋ 80 ＋ c ＋ d ＋ 64 ＋ 8192 ＋ g ＋ 80 ＋ i ＋ 64 ＋ 8192 ＋ 64 ＋ n ＋ 16 ＋ t ＋ u ＋ 16)<br>× pd_lck_pool_partition の値<br>c：pd_lck_hash_entry を省略，又は 0 を指定している場合：<br>（↓（8 ＋ 8 × MAX（((p ＋ 3）×（pd_max_access_tables の値＋ 4 ＋ 5 × 2）<br>＋↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 4）<br>÷ 10，11261)）÷ 16 ↓＋ 1）× 16<br>pd_lck_hash_entry に 2 以上の素数でない値を指定している場合：<br>（↓（8 ＋ 8 × pd_lck_hash_entry の値を超えない最大の素数）÷ 16 ↓＋ 1）<br>× 16<br>pd_lck_hash_entry に 1，又は素数を指定している場合：<br>（↓（8 ＋ 8 × pd_lck_hash_entry の値）÷ 16 ↓＋ 1）× 16<br>d：((p ＋ 3）×（pd_max_access_tables の値＋ 4 ＋ 5 × 2）＋<br>↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 4）× 128<br>g：pd_utl_exec_mode の値 =1，かつ p ＞ 32 の場合：<br>((p ＋ 3）× 3 ＋ p）× 320<br>pd_utl_exec_mode の値 =0，又は p ≦ 32 の場合：<br>((p ＋ 3）× 3 ＋ 32）× 320 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| | i：pd_utl_exec_mode の値 =1，かつ p ＞ 32 の場合：<br>((↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 5<br>＋((p ＋ 3) ×（pd_max_access_tables の値＋ 4)) × 2)<br>＋↓（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓）÷ 3 ↓<br>＋ p ×（pd_max_rdarea_no の値＋ 1）＋（p ＋ 3）× 2 × 2 × 5)<br>を偶数に切り上げた値× 112<br>pd_utl_exec_mode の値 =0，又は p ≦ 32 の場合：<br>((↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 5<br>＋((p ＋ 3) ×（pd_max_access_tables の値＋ 4)) × 2)<br>＋↓（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓）÷ 3 ↓<br>＋ 32 ×（pd_max_rdarea_no 値＋ 1）＋（p ＋ 3）× 2 × 2 × 5)<br>を偶数に切り上げた値× 112<br><br>n：pd_utl_exec_mode の値 =1，かつ p ＞ 32 の場合：<br>((p ＋ 3) × 3 ＋ p) × 80<br>pd_utl_exec_mode の値 =0，又は p ≦ 32 の場合：<br>((p ＋ 3) × 3 ＋ 32) × 80<br>p：pd_max_users の値<br>t：pd_utl_exec_mode の値 =1，かつ p ＞ 32 の場合：<br>32 ＋((p ＋ 3) × 3 ＋ p) ×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 4<br>pd_utl_exec_mode の値 =0，又は p ≦ 32 の場合：<br>32 ＋((p ＋ 3) × 3 ＋ 32) ×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 4<br><br>u：pd_utl_exec_mode の値 =1，かつ p ＞ 32 の場合：<br>32 ＋((↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 5<br>＋((p ＋ 3) ×（pd_max_access_tables の値＋ 4)) × 2)<br>＋↓（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓）÷ 3 ↓<br>＋ p ×（pd_max_rdarea_no の値＋ 1）＋（p ＋ 3）× 2 × 2 × 5)<br>を偶数に切り上げた値×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 4<br>pd_utl_exec_mode の値 =0，又は p ≦ 32 の場合：<br>32 ＋((↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 5<br>＋((p ＋ 3) ×（pd_max_access_tables の値＋ 4)) × 2)<br>＋↓（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓）÷ 3 ↓<br>＋ 32 ×（pd_max_rdarea_no の値＋ 1）＋（p ＋ 3）× 2 × 2 × 5)<br>を偶数に切り上げた値×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 4 |
| トランザクションサーバ | 304 ＋ 32 ＋ 192 × m × 2 ＋ 1048<br>＋（416 ＋ 720 ＋ 256 ＋ 392 × 2 ＋ 128）×（m × 2 ＋ 7）＋ 256 × 2<br>m：pd_max_users の値 |
| タイマサーバ | 32 ×（pd_max_users の値＋ 3）×（1 ＋ユニット内ユティリティサーバ数＋ 1）<br>＋ 1440 ＋（48 － 32）× 2<br><br>ユニット内ユティリティサーバ数は 26 ＋ 12 |
| 統計ログサーバ | 424 ＋ 128 × 16 ＋ 32 ＋ 288 × 2 ＋ 1168 ＋ 144 × 3<br>＋ pd_stj_buff_size の値× 1024 × 3 ＋ 64 ＋ 4096 ＋ 8192 |
| プロセスサーバ | 176 ＋ 528 × a ＋ 80 ＋ 256 ＋（pd_max_server_process の値＋ 50）×（256 ＋ 160）＋ 16 ＋ 8 × 16 ＋ 16 ＋ 16 ＋ 64 ＋ 64 ×（pd_module_trace_max の値＋ 1）<br><br>a：シングルサーバの場合：125 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| システムマネジャ | 672 ＋（48 ＋ 8）× a × 2 ＋（112 ＋ 8）×（b ＋ 30 ＋ 2）＋（104 ＋ 8）×（c ＋ 1）＋ 40 × b × 14 ＋ 256 ＋ 256 ＋ 40 × d ＋ 16 × e ＋ 8 ＋ 5864 ＋ 236 ＋ f ＋ 16 ＋ 1024 ＋ 272 × h<br><br>a：シングルサーバの定義数<br>b：ユニット内のシングルサーバ数<br>c：ユニット数<br>d：pdunit オペランドの -c オプションの指定数<br>e：pdcltgrp オペランド指定数<br>f：2056 ＋ 128 ×（g ＋ 3）<br>g：シングルサーバの場合：92<br>h：pd_security_host_group オペランドに指定したホストに対応する IP アドレスの数<br>pd_security_host_group オペランドを指定していない場合は 0 |
| ネームサーバ | 169984 |
| ノードマネジャ | ↑（1312 ＋ 320 × HiRDB システムの全ユニット数<br>＋ 96 × HiRDB システムの全サーバ数＋ 2048 ＋ 960<br>＋ 72 ×ユニット内 HiRDB サーバ数＋ 240 × A ＋ 44 × A ＋ 28 × A ＋ 16 × B ＋ 48）<br>÷ 1024 ↑× 1024<br>A：pd_utl_exec_mode = 0 の場合：1024<br>pd_utl_exec_mode = 1 で，ユニット内にシングルサーバがある場合：pd_max_users の値× 10 ＋ 400<br>pd_utl_exec_mode = 1 で，ユニット内にシングルサーバがない場合：pd_max_users の値× 7<br>なお，A の値が 1024 を超えない場合は，A を 1024 に置き換えて計算してください。<br>B：pdcltgrp オペランドを指定しない場合：0<br>pdcltgrp オペランドを指定する場合：pdcltgrp オペランドの指定数＋ 1 |
| I/O サーバ | ↑（56 ＋（↑（56 ＋ A）÷ 32 ↑× 32））÷ 128 ↑× 128<br>**A：pd_large_file_use=N を指定した場合**<br>3248 ＋（14 ＋ 16）× 808 ＋ 1 × 272 ＋ 534 × 272 ＋ 16 × 272 ＋ pd_max_file_no 値× 808 ＋（48 － 32）× 3<br>**pd_large_file_use=Y（又は省略）を指定した場合**<br>3248 ＋（14 ＋ 16）× 972 ＋ 1 × 276 ＋ 534 × 276 ＋ 16 × 276 ＋ pd_max_file_no 値× 972 ＋（48 － 32）× 3 |
| ログサーバ | 32 ＋ 48 ＋ 128 × 19 ＋ 432 ＋ 128 × 7 ＋ 1168 ＋ 512<br>＋↑（128 ＋ 256 ＋ 160 ＋ 8 ＋ 64）÷ pd_log_rec_leng の値↑× pd_log_rec_leng の値<br>＋ 64 ＋ 4096 × 2 ＋（768 ＋ 512）× B<br>＋ 144 × pd_log_write_buff_count の値<br>＋（pd_log_write_buff_count の値＋ A）<br>×↑｛pd_log_max_data_size の値＋（68 ＋ 44 ＋ 96 ＋ 160)｝÷ 4096 ↑× 4096<br>＋ C ＋↑｛512 ＋ 256 ＋ 128 × B ＋ 464 × B｝÷（8192 － 128）↑× 8192<br><br>A：16<br>B：pdlogadfg -d sys オペランドの指定数<br>C：512 |
| シンクポイントダンプサーバ | ↑（384 ＋ 1536 × 2）÷ 1024 ↑× 1024<br>＋↑｛(128 ＋ 80 ＋ 240 ＋ 240）＋ 192 ×（pdlogadfg -d spd の指定数）<br>＋ 416 ×（pdlogadpf -d spd の指定数）＋ 1023｝<br>÷ 1024 ↑× 1024 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| ユニット共通 | a ＋ b ＋ 80 ＋（m ＋ 3）× c ＋ 64 ＋ 48 ＋ d ＋ e<br>＋（pd_max_server_process の値＋ 100）×（64 ＋ 16）＋ 32<br>＋（pd_max_server_process の値＋ 100 ＋ 384）× 32 ＋ 32 ＋ f ＋ g ＋ h ＋ i<br>＋（pd_max_server_process の値＋ 127）× 48 ＋ 32<br><br>a：34768 ＋ p × 4<br>b：3480<br>c：2760<br>d：48 × 32<br>e：80 ＋ 96 ×｛(m ＋ 3）× 2<br>＋ MAX（5, ↓［m ＋ 3］÷ 10 ↓）＋ 7｝<br>f：512 ×（13 ＋ 3）× 2<br>g：｛(128 ＋ pd_lck_until_disconnect_cnt の値× 112 ＋ 4095）÷ 4096｝<br>× 4096 × 2<br>h：↑（pd_registered_port で指定したポート番号の数× 16 ＋ 32 ＋ 1023）<br>÷ 1024 ↑× 1024<br>pd_registered_port を指定していない場合は 0<br>i：k が 2 以上の場合は 32 ＋（8 ＋ 8 × k）× n<br>それ以外の場合は 0<br>k：pd_lck_pool_partition の値<br>m：pd_max_users の値<br>n：(m+3）× 2 ＋ MAX｛5, ↓（m ＋ 3）÷ 10 ↓｝＋ 7<br>p：pd_dbbuff_modify に Y を指定している場合：16 ＋ pd_max_add_dbbuff_shm_no の値＋ pd_max_resident_rdarea_shm_no の値<br>上記以外：16 ＋ pd_max_resident_rdarea_shm_no の値 |
| トランザクションログサーバ | 1168 ＋ 688 × A<br>＋｛<br>128 × B ＋ 144<br>＋［G ＋↑（128 ＋ 256 ＋ 8 ＋ 224）÷ pd_log_rec_leng の値↑× pd_log_rec_leng の値<br>＋↑（pd_log_max_data_size の値＋ 68 ＋ 44 ＋ 96 ＋ 160）<br>÷ pd_log_rec_leng の値↑× pd_log_rec_leng の値］× D<br>＋ E ＋（48 ＋ 8）× B × 2<br>｝×ユニット内のサーバ数<br>＋ 600 × B ＋ 128 × B ＋ 64 × B × C ＋ 144<br>＋｛<br>G ＋↑（128 ＋ 256 ＋ 8 ＋ 224）÷ pd_log_rec_leng の値↑× pd_log_rec_leng の値<br>＋↑（pd_log_max_data_size の値＋ 68 ＋ 44 ＋ 96 ＋ 160）<br>÷ pd_log_rec_leng の値↑× pd_log_rec_leng の値<br>｝<br>＋ E ＋（48 ＋ 8）×（B × 2 ＋ 2）<br><br>A：2<br>B：7 ＋ pd_max_users の値× 2<br>C：1<br>D：シングルサーバの場合，pd_log_rollback_buff_count の値が 0 のときは 8,<br>それ以外のときは pd_log_rollback_buff_count の値<br>E：512<br>G：512 |
| ステータスサーバ | ↑ 64 ÷ 32 ↑× 32 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| 監査証跡管理サーバ | ↑ A ÷ 1024 ↑× 1024<br>A：pd_aud_file_name オペランドの指定がない場合は 704<br>pd_aud_file_name オペランドの指定がある場合は 704 ＋（320 × 200）+B ＋ C<br>B：pd_aud_async_buff_size オペランドの値が 0 の場合は 0<br>pd_aud_async_buff_size オペランドの値が 4096 以上の場合は次の計算値<br>（176 × pd_aud_async_buff_count オペランドの値）<br>+ {(↑ pd_aud_async_buff_size オペランドの値÷ 4096 ↑× 4096）<br>× pd_aud_async_buff_count オペランドの値} ＋ 4096<br>C：pd_aud_auto_loading オペランドに N を指定している場合は 0<br>pd_aud_auto_loading オペランドに Y を指定している場合は 256 × 2 ＋ 256 |

## 15.1.4 シングルサーバが使用する共用メモリの計算式

シングルサーバが使用する共用メモリの計算式を次に示します。

**32 ビットモードの場合**（単位：キロバイト）

計算式 1 ＋↑ {(↑（40 ＋（計算式 2 ～計算式 7 を加算した値））÷ 512 ↑× 512)} ÷ 1024 ↑

**64 ビットモードの場合**（単位：キロバイト）

計算式 1 ＋↑ {(↑（72 ＋（計算式 2 ～計算式 7 を加算した値））÷ 512 ↑× 512)} ÷ 1024 ↑

**計算式 1 ～ 7 についての注意事項**

- pd_rdarea_open_attribute_use オペランドに Y を指定する場合に計算式 3 を加算します。
- pd_dbsync_point 又は pd_system_dbsync_point オペランドのどちらかの指定が commit の場合に計算式 4 を加算します。pd_system_dbsync_point オペランドの省略値は commit です。
- pd_dfw_awt_process オペランドを指定する場合に計算式 5 を加算します。
- pd_sds_shmpool_size オペランドを省略すると，次の値が設定されます。
  32 ビットモードの場合：
  ↑ {(↑（40 ＋（計算式 2 ～計算式 7 の合計値））÷ 512 ↑× 512)} ÷ 1024 ↑
  64 ビットモードの場合：
  ↑ {(↑（72 ＋（計算式 2 ～計算式 7 の合計値））÷ 512 ↑× 512)} ÷ 1024 ↑
- pd_max_commit_write_reclaim_no オペランドを省略した場合（ただし，pd_sysdef_default_option オペランドに ” v6compatible ” 又は ” v7compatible ” を指定した場合は除きます），又は pd_max_commit_write_reclaim_no オペランドに 0 以外を指定した場合に計算式 7 を加算します。ただし，計算式 4 を加算している場合には加算する必要はありません。
- pd_max_resident_rdarea_no オペランドを指定する場合に計算式 7 を加算します。

計算式 1 ～ 7 を次に示します。

| 計算式の種類 | 共用メモリの計算式 |
|---|---|
| 計算式 1<br>（単位：キロバイト） | **● 32 ビットモードの場合**<br>b × 1.3 ＋ c ＋ d ＋ f ＋ 1.6 ＋ q ＋ r ＋ 4<br>＋ {［(a ＋ 12) ÷ 13］ × 1.1 ＋ ［(a ＋ 62) ÷ 63］ ＋ 3.7} × (e ＋ 3) ＋ 3.5<br>＋ {<br>↑ (↑ (b ÷ 64) ↑) × (8 ÷ 16) ↑ × 4 × 4<br>＋ 12 × {(b ÷ 3) ＋ 1 － mod (b ÷ 3, 2)}<br>＋ 8 × a × {(e ＋ 3) × 2 ＋ 1 ＋ MAX (e ÷ 10, 5)} ＋ 32 ＋ 20000<br>＋↑ {(c ÷ 8) ＋ 7} ÷ 64 ↑ × 8 ＋↑ {(f ÷ 8) ＋ 7} ÷ 64 ↑ × 8<br>＋ MAX {a × (e ＋ 3), c ÷ 8} × 104 ＋ MAX {a × (e ＋ 3), f ÷ 8} × 24<br>＋↑ {(q ÷ 4) ＋ 7} ÷ 64 ↑ × 8<br>＋↑ {[ (r － (s × 592 ＋ t × 916 ＋ u × 172)) ÷ 2] ＋ 7} ÷ 64 ↑ × 8<br>＋ MAX {13 × (e ＋ 3), q ÷ 4} × 88<br>＋ MAX {21 × (e ＋ 3), [r － (s × 592 ＋ t × 916 ＋ u × 172) ] ÷ 2} × 60<br>＋ 44 ＋ 256 ＋ 1024 ＋ 512 ※1<br>} ÷ 1024 ＋ y ＋ 7.5<br><br>＋↑ {248 × v × w ＋ 47 × v ＋ 72} ÷ 1024 ↑<br>＋↑ {↑ (28 ＋ (↑ (32 ＋ ((↑ g ÷ 127 ↑ ＋ 1) × 2048 ＋ 128)) ÷ 32 ↑ × 32))<br>÷ 128 ↑ × 128<br>} ÷ 1024 ↑<br>D<br>＋ Σ (Ei)<br>i=1<br>pd_def_buf_control_area_assign **オペランドに** INITIAL **を指定するか，又はこのオペランドを省略した場合に加算します。**<br>＋ {［(a ＋ 12) ÷ 13］ × 1.1 ＋ ［(a ＋ 62) ÷ 63］ ＋ 3.7} × (e ＋ 7)<br><br>**● 64 ビットモードの場合**<br>b × 1.3 ＋ c ＋ d ＋ f ＋ 1.6 ＋ q ＋ r ＋ 5<br>＋ {［(a ＋ 12) ÷ 13］ × 1.2 ＋ ［(a ＋ 62) ÷ 63］ × 1.5 ＋ 4.1}<br>× (e ＋ 3) ＋ 3.5<br>＋ {<br>↑ (↑ (b ÷ 64) ↑) × (8 ÷ 16) ↑ × 4 × 4 ＋ 12<br>× {(b ÷ 3) ＋ 1 － mod (b ÷ 3, 2)}<br>＋ 8 × a × {(e ＋ 3) × 2 ＋ 1 ＋ MAX (e ÷ 10, 5)} ＋ 48 ＋ 20000<br>＋↑ {(c ÷ 8) ＋ 7} ÷ 64 ↑ × 8 ＋↑ {(f ÷ 8) ＋ 7} ÷ 64 ↑ × 8<br>＋ MAX {a × (e ＋ 3), c ÷ 8} × 104 ＋ MAX {a × (e ＋ 3), f ÷ 8} × 40<br>＋↑ {(q ÷ 4) ＋ 7} ÷ 64 ↑ × 8<br>＋↑ {[ (r － (s × 600 ＋ t × 936 ＋ u × 184)) ÷ 2] ＋ 7} ÷ 64 ↑ × 8<br>＋ MAX {13 × (e ＋ 3), q ÷ 4} × 104<br>＋ MAX {21 × (e ＋ 3), [r － (s × 600 ＋ t × 936 ＋ u × 184) ] ÷ 2} × 72<br>＋ 72 ＋ 256 ＋ 1536 ＋ 512 ※1<br>} ÷ 1024 ＋ y ＋ 7.5<br><br>＋↑ {248 × v × w ＋ 64 × v ＋ 72} ÷ 1024 ↑<br>＋↑ {↑ (56 ＋ (↑ (56 ＋ ((↑ g ÷ 127 ↑ ＋ 1) × 2048 ＋ 128)) ÷ 32 ↑ × 32))<br>÷ 128 ↑ × 128<br>} ÷ 1024<br>D<br>＋ Σ (Ei)<br>i=1<br>pd_def_buf_control_area_assign **オペランドに** INITIAL **を指定するか，又はこのオペランドを省略した場合に加算します。**<br>＋ {［(a ＋ 12) ÷ 13］ × 1.2 ＋ ［(a ＋ 62) ÷ 63］ × 1.5 ＋ 4.1} × (e ＋ 7) |

| 計算式の種類 | 共用メモリの計算式 |
|---|---|
| 計算式 2<br>（単位：バイト） | ● 32 ビットモードの場合<br>500 × 1024<br>＋ 5072 ×（e ＋ 15）＋（↑ 372 × g ÷ 16 ↑× 16）＋ 48 ※1 × g ＋ 328 × h<br>＋ 112 ×（p ＋ 240）※5<br>＋ 96 × x ＋ 32 × j ＋ 132 × {19 ＋（e ＋ 3）× 3}<br>＋ 48 × n ＋ 48 × {(e ＋ 3) × 2 ＋ 1 ＋ MAX（5,（e ＋ 3）÷ 10)}<br>＋ 68 × B ＋ 144 × A ＋ 80 ＋ 32 × g ＋ 64 ※2 ＋ 368 ※3<br>＋（(↓（↑（g ÷ 8）↑＋ 3）÷ 4 ↓）× 4）× j<br>● 64 ビットモードの場合<br>500 × 1024<br>＋ 9416 ×（e ＋ 15）＋（↑ 472 × g ÷ 16 ↑× 16）<br>＋（↑ 56 ※1 × g ÷ 16 ↑× 16）＋ 344 × h<br>＋（↑ 136 ×（p ＋ 240）÷ 16 ↑× 16）※6<br>＋ 144 × x ＋ 48 × j ＋ 240 × {19 ＋（e ＋ 3）× 3}<br>＋ 64 × n ＋ 96 × {(e ＋ 3) × 2 ＋ 1 ＋ MAX（5,（e ＋ 3）÷ 10)}<br>＋ 68 × B ＋ 160 × A ＋ 96 ＋ 48 × g ＋ 64 ※2 ＋ 448 ※3<br>＋（(↓（↑（g ÷ 8）↑＋ 7）÷ 8 ↓）× 8）× j |
| 計算式 3<br>（単位：バイト） | ● 32 ビットモードの場合<br>{[（↑ ↑ g ÷ 8 ↑÷ 4 ↑）× 4] ＋ 8} × {(e ＋ 3) × 2 ＋ 12}<br>● 64 ビットモードの場合<br>{[（↑ ↑ g ÷ 8 ↑÷ 8 ↑）× 8] ＋ 8} × {(e ＋ 3) × 2 ＋ 12} |
| 計算式 4<br>（単位：バイト） | ● 32 ビットモードの場合<br>（32 ＋ 16 × x）×（e × 2 ＋ 7 ＋ 1）＋ 16<br>● 64 ビットモードの場合<br>（48 ＋ 32 × x）×（e × 2 ＋ 7 ＋ 1）＋ 16 |
| 計算式 5<br>（単位：バイト） | ● 32 ビットモードの場合<br>72 ＋ 52 × C ＋ 68 × x<br>pd_dbbuff_trace_level オペランドに 1 を指定する場合に加算します。<br>＋ 320 × x<br>● 64 ビットモードの場合<br>96 ＋ 56 × C ＋ 72 × x<br>pd_dbbuff_trace_level オペランドに 1 を指定する場合に加算します。<br>＋ 640 × x |
| 計算式 6<br>（単位：バイト） | ● 32 ビットモードの場合<br>（32 ＋ 16 × x）× F ＋ 16<br>● 64 ビットモードの場合<br>（48 ＋ 32 × x）× F ＋ 16 |
| 計算式 7<br>（単位：バイト） | 16 ＋ 112 ＋（48 ＋ 48 × G）＋（48 ＋ 32 × H） |

a：pd_max_access_tables オペランドの値

b：pd_sql_object_cache_size オペランドの値

c：pd_table_def_cache_size オペランドの値

d：pd_auth_cache_size オペランドの値

e：pd_max_users オペランドの値

f：pd_view_def_cache_size オペランドの値

g：pd_max_rdarea_no オペランドの値

h：pd_max_file_no オペランドの値

i：サーバ内のインデクス総数
ただし，分割キーインデクスの場合はサーバ内の分割数になります。

j：インデクス用のグローバルバッファプール数
pd_dbbuff_modify オペランドに Y を指定している場合，pdbuffer コマンドの指定数に，
pd_max_add_dbbuff_no オペランドの値を加算して計算します。

n：pd_lck_until_disconnect_cnt オペランドの値

p：pd_assurance_index_no オペランドの値

q：pd_type_def_cache_size オペランドの値

r：pd_routine_def_cache_size オペランドの値

s：インストールしたプラグインの数

t：DML で使用するプラグイン関数の総数 ※4

u：DML で使用するプラグイン関数のパラメタ総数 ※4

v：pd_max_list_users オペランドの値

w：pd_max_list_count オペランドの値

x：総グローバルバッファ数（pdbuffer オペランドの指定数）
pd_dbbuff_modify オペランドに Y を指定している場合，pdbuffer コマンドの指定数に，
pd_max_add_dbbuff_no オペランドの値を加算して計算します。

y：pd_registry_cache_size オペランドの値

A：pd_assurance_table_no オペランドの値

B：サーバ内の最大トランザクション数（2 × e ＋ 7）

C：pd_dfw_awt_process オペランドの値

D：指定した pdplgprm オペランドの総数

Ei：i 番目の pdplgprm オペランドで指定した共用メモリのサイズ

F：pd_max_commit_write_reclaim_no オペランドの値

G：pd_max_resident_rdarea_no オペランドの値

H：pd_max_resident_rdarea_shm_no オペランドの値

注※ 1
pd_max_list_users 及び pd_max_list_count オペランドの両方とも 0 でない場合に加算します。

注※ 2
pd_max_ard_process オペランドに 1 以上を指定した場合に加算します。

注※ 3

再編成時期予測機能を使用する場合に加算します。

注※ 4

DML で使用するプラグイン関数の総数及び DML で使用するプラグイン関数のパラメタの総数は，次に示す SQL で求められます。

```
SELECT COUNT(*),SUM(N_PARAM) FROM MASTER.SQL_PLUGIN_ROUTINES
WHERE PLUGIN_NAME = 'プラグイン名称'
AND (TIMING_DESCRIPTOR = 'ADT_FUNCTION'
    OR TIMING_DESCRIPTOR IS NULL
    OR TIMING_DESCRIPTOR = 'BEFORE_INSERT'
    OR TIMING_DESCRIPTOR = 'AFTER_INSERT'
    OR TIMING_DESCRIPTOR = 'BEFORE_UPDATE'
    OR TIMING_DESCRIPTOR = 'AFTER_UPDATE'
    OR TIMING_DESCRIPTOR = 'BEFORE_DELETE'
    OR TIMING_DESCRIPTOR = 'AFTER_DELETE'
    OR TIMING_DESCRIPTOR = 'BEFORE_PURGE_TABLE'
    OR TIMING_DESCRIPTOR = 'AFTER_PURGE_TABLE'
    OR TIMING_DESCRIPTOR = 'INDEX_SEARCH'
    OR TIMING_DESCRIPTOR = 'INDEX_COUNT'
    OR TIMING_DESCRIPTOR = 'INDEX_INSERT'
    OR TIMING_DESCRIPTOR = 'INDEX_BEFORE_UPDATE'
    OR TIMING_DESCRIPTOR = 'INDEX_AFTER_UPDATE'
    OR TIMING_DESCRIPTOR = 'INDEX_DELETE'
    OR TIMING_DESCRIPTOR = 'PURGE_INDEX'
    OR TIMING_DESCRIPTOR = 'INDEX_MAINTENANCE_DEFERRED'
    OR TIMING_DESCRIPTOR = 'BEFORE_INSERT_DC'
    OR TIMING_DESCRIPTOR = 'BEFORE_UPDATE_DC'
    OR TIMING_DESCRIPTOR = 'BEFORE_DATA_CHECK'
    OR TIMING_DESCRIPTOR = 'AFTER_DATA_CHECK')
```

注※ 5

pd_sysdef_default_option オペランドに ” v6compatible ” 又は ” v7compatible ” を指定した場合，112 ×（p ＋ 240）の替わりに 112 × MAX（p，i × 1.2）で計算してください。

注※ 6

pd_sysdef_default_option オペランドに ” v6compatible ” 又は ” v7compatible ” を指定した場合，（↑ 136 ×（p ＋ 240）÷ 16 ↑× 16）の替わりに（↑ 136 × MAX（p，（i × 1.2））÷ 16 ↑× 16）で計算してください。

## 15.1.5 グローバルバッファが使用する共用メモリの計算式

グローバルバッファが使用する共用メモリサイズは，pdbuffer 文ごとに計算式 1 で求めます。
pd_dbbuff_modify オペランドに Y を指定している場合は，計算式 2 を加算します。計算式 1 及び 2 で求めた値を合計した値が，グローバルバッファが使用する共用メモリの所要量です。

なお，pdbuffer オペランドを省略した場合，HiRDB が共用メモリサイズを自動計算するので見積もりは必要ありません。

### (1) グローバルバッファが使用する共用メモリを実メモリ上に固定しない場合

グローバルバッファが使用する共用メモリを実メモリ上に固定しない場合（pd_dbbuff_attribute オペランドに free を指定した場合）は，次の計算式で求めます。

| 計算式の種類 | 共用メモリの計算式（単位：キロバイト） |
|---|---|
| 計算式 1 | ●32 ビットモードの場合<br>n<br>Σ｛《管理領域部》＋《データ格納部》｝÷ 1024<br>i=1<br>《管理領域部》：<br>↑｛736 ＋ 64 ＋（296 ＋ 64※1）×（Pi ＋ 4）<br>＋（124 ＋ 80※2＋ 96 × A × Mi）× Ui｝÷ 4096 ↑× 4096<br>《データ格納部》：<br>Si ×｛Pi ＋ 4 ＋（Ui × Mi × A)｝<br>●64 ビットモードの場合<br>n<br>Σ｛《管理領域部》＋《データ格納部》｝÷ 1024<br>i=1<br>《管理領域部》：<br>↑｛928 ＋ 64 ＋（480 ＋ 112※1）×（Pi ＋ 4）<br>＋（176 ＋ 96※2＋ 136 × A × Mi）× Ui｝÷ 4096 ↑× 4096<br>《データ格納部》：<br>Si ×｛Pi ＋ 4 ＋（Ui × Mi × A)｝ |
| 計算式 2 | ●32 ビットモードの場合<br>{↑［((s × 1024 ÷ 4）÷ 8）＋ 112］÷ 4096 ↑× 4096 ×↑ a ÷（s × 1024）↑<br>}<br>●64 ビットモードの場合<br>{↑［((s × 1024 ÷ 4）÷ 8）＋ 144］÷ 4096 ↑× 4096 ×↑ a ÷（s × 1024）↑<br>} |

n：グローバルバッファプール数

i：計算対象のグローバルバッファプール定義

P：グローバルバッファ面数

A：非同期 READ 機能を使用する場合は 2，使用しない場合は 1

M：一括入力最大ページ数
pd_max_ard_process オペランドに 1 以上を指定した場合は指定値の 2 倍です。

U：同時実行最大プリフェッチ数

S：グローバルバッファに割り当てた RD エリアの最大ページ長

s：SHMMAX 指定値

a：計算式 1 の総計

注※ 1　LOB 用グローバルバッファの場合に加算します。

注※ 2　pd_max_ard_process オペランドに 1 以上を指定した場合に加算します。

## （2）グローバルバッファが使用する共用メモリを実メモリ上に固定する場合

グローバルバッファが使用する共用メモリを実メモリ上に固定する場合（pd_dbbuff_attribute オペランドに fixed を指定した場合）は，次の計算式で求めます。

| 計算式の種類 | 共用メモリの計算式（単位：キロバイト） |
|---|---|
| 計算式 1 | ● 32 ビットモードの場合<br>$\sum_{i=1}^{n}$ {↑ {《管理領域部》+《データ格納部》} ÷ p ↑× p} ÷ 1024<br>《管理領域部》:<br>↑ {↑ {736 ＋ 64 ＋（296 ＋ 64※1）×（Pi ＋ 4）<br>＋（124 ＋ 80※2 ＋ 96 × A × Mi）× Ui} ÷ 4096 ↑× 4096<br>《データ格納部》<br>Si × {Pi ＋ 4 ＋（Ui × Mi × A)}<br>● 64 ビットモードの場合<br>$\sum_{i=1}^{n}$ {↑ {《管理領域部》＋《データ格納部》} ÷ p ↑× p} ÷ 1024<br>《管理領域部》:<br>↑ {↑ {928 ＋ 64 ＋（480 ＋ 112※1）×（Pi ＋ 4）<br>＋（176 ＋ 96※2 ＋ 136 × A × Mi）× Ui} ÷ 4096 ↑× 4096<br>《データ格納部》:<br>Si × {Pi ＋ 4 ＋（Ui × Mi × A)} |
| 計算式 2 | ● 32 ビットモードの場合<br>{↑ {↑［((s × 1024 ÷ 4) ÷ 8) ＋ 112］÷ 4096 ↑× 4096 ×↑ a ÷（s × 1024）↑} ÷ p ↑× p}<br>● 64 ビットモードの場合<br>{↑ {↑［((s × 1024 ÷ 4) ÷ 8) ＋ 144］÷ 4096 ↑× 4096 ×↑ a ÷（s × 1024）↑} ÷ p ↑× p} |

n：グローバルバッファプール数

i：計算対象のグローバルバッファプール定義

P：グローバルバッファ面数

A：非同期 READ 機能を使用する場合は 2，使用しない場合は 1

M：一括入力最大ページ数
pd_max_ard_process オペランドに 1 以上を指定した場合は指定値の 2 倍です。

U：同時実行最大プリフェッチ数

S：グローバルバッファに割り当てた RD エリアの最大ページ長

p：Windows の Large Page でのページサイズ
pdntenv コマンドで確認できます。

s：SHMMAX 指定値

a：計算式 1 の総計

注※ 1　LOB 用グローバルバッファの場合に加算します。

注※ 2　pd_max_ard_process オペランドに 1 以上を指定した場合に加算します。

## 15.1.6　SQL 実行時に必要なメモリ所要量の計算式

### (1) グループ分け高速化機能実行時に必要なメモリ所要量の求め方

クライアント環境定義で PDSQLOPTLVL オペランドを指定するか，HiRDB システム定義で

pd_optimize_level オペランドを指定（又は省略）した場合，適用条件を満たす SQL を実行すると，グループ分け高速化機能が働きます。このとき，HiRDB はクライアント環境定義の PDAGGR オペランドの値に基づいてプロセス固有メモリを確保します。確保するメモリサイズを次に示します。

**計算式**

```
e＋↑ d÷4 ↑×4＋↑（17＋4×a＋4×b＋c＋d）÷4 ↑×4×（N＋1）
                                                （単位：バイト）
```

a：グループ化する列の数

b：集合関数の演算数
COUNT，SUM，MAX，MIN は一つにつき 1 で換算します。
AVG（COUNT），AVG（SUM）は一つにつき 2 で換算します。

c：グループ化する列の列長（表 15-4 を参照して求めてください）

d：集合関数の演算領域長（表 15-4 を参照して求めてください）

e：32 ビットモードの場合：MAX（4 × N × 24 , 16408）
64 ビットモードの場合：MAX（8 × N × 40 , 32808）

N：クライアント環境定義の PDAGGR オペランドの値

**表 15-4 グループ化するときの列の長さ及び集合関数の演算領域の長さ**

| 列のデータ型 | グループ化する列の列長 | 集合関数の演算領域長※1 |
|---|---|---|
| INTEGER | 4 | 6 |
| SMALLINT | 2 | 4※2 |
| DECIMAL(p,s) | ↑（p＋1）÷2 ↑ | ↑（p＋7）÷2 ↑※3 |
| FLOAT | 8 | 10 |
| SMALLFLT | 4 | 6 |
| INTERVAL YEAR TO DAY | 5 | 8 |
| INTERVAL HOUR TO SECOND | 4 | 6 |
| CHAR(n) | n | n＋3 |
| VARCHAR(n) | n＋2 | n＋5 |
| NCHAR(n) | 2×n | 2×n＋2 |
| NVARCHAR(n) | 2×n＋2 | 2×n＋4 |
| MCHAR(n) | n | n＋3 |
| MVARCHAR(n) | n＋2 | n＋5 |
| DATE | 4 | 6 |
| TIME | 3 | 6 |
| BLOB(n) | － | － |
| BINARY(n) | n＋2 | n＋5 |

（凡例）－：該当しません。

注※1

集合関数が COUNT の場合，集合関数演算領域長はデータ型にかかわらず 6 になります。

注※ 2

集合関数が AVG，SUM の場合は集合関数演算領域長は 6 になります。

注※ 3

集合関数が AVG，SUM の場合，集合関数演算領域長は次の値になります。
集合関数の値の型が DECIMAL 型で精度が 29 けたのとき：18
集合関数の値の型が DECIMAL 型で精度が 38 けたのとき：23
集合関数の値のデータ型の規則については，マニュアル「HiRDB Version 8 SQL リファレンス」の「集合関数」を参照してください。

## (2) 列ごとのデータ抑制指定時に必要なメモリ所要量の求め方

列ごとのデータ抑制を指定（CREATE TABLE の列定義に SUPPRESS を指定）した表に対してアクセスするときに使用するメモリサイズは，次に示す計算式で求められます。

**計算式**

a ＋ 128　（単位：バイト）

ａ：表中で列ごとのデータ抑制が指定されている列の定義長の合計値

## (3) ハッシュジョイン及び副問合せのハッシュ実行時に必要なメモリ所要量の求め方

クライアント環境定義で PDADDITIONALOPTLVL オペランドを指定するか，HiRDB システム定義で pd_additional_optimize_level オペランドを指定すると，SQL 拡張最適化オプションが働きます。この SQL 拡張最適化オプションで，「ハッシュジョイン，副問合せのハッシュ実行の適用（APPLY_HASH_JOIN）」を指定した場合，表の結合又は副問合せの SQL を実行すると，次に示すメモリサイズのプロセス固有領域を確保します。

**計算式**

**● 32 ビットモードの場合**

$$\sum_{i=1}^{a}(13 \times 1024 + 6 \times 1024 \times b + c)$$

**● 64 ビットモードの場合**

$$\sum_{i=1}^{a}(13 \times 1024 + 7 \times 1024 \times b + c)$$

（単位：バイト）

ａ：SELECT 文のハッシュジョイン最大数
SELECT 文のハッシュジョイン最大数については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

ｂ：ハッシュ表行数によって適用されるハッシュジョイン処理を求めて，次に示す表から代入する値を決定してください。

| ハッシュ表行数の目安 | 適用されるハッシュジョイン処理 | b の値 |
|---|---|---|
| 1500 以内 | 一括ハッシュジョイン | 0.5 |

| ハッシュ表行数の目安 | 適用されるハッシュジョイン処理 | | b の値 |
|---|---|---|---|
| 1500 ×（バケット分割数÷ 3）以内 | バケット分割ハッシュジョイン | 1 レベルバケット分割 | 1 |
| 1500 ×（バケット分割数÷ 3）$^2$ 以内 | | 2 レベルバケット分割 | 2 |
| 1500 ×（バケット分割数÷ 3）$^2$ を超える場合 | | 3 レベルバケット分割 | 3 |

**ハッシュ表行数：**ジョインの場合はジョインの内表件数です。副問合せの場合は探索条件中の外への参照列を含む述語を除いた副問合せ探索件数です。

**バケット分割数：**MIN｛↓（ハッシュ表サイズ÷ 2）÷ハッシュ表ページ長↓，64｝

**ハッシュ表サイズ：**HiRDB システム定義の pd_hash_table_size オペランド，又はクライアント環境変数の PDHASHTBLSIZE オペランドで指定した値です。

**ハッシュ表ページ長：**次に示す表から c（ハッシュ表最大行長）に対応するハッシュ表ページ長を選択してください。

| ハッシュ表最大行長 | ハッシュ表ページ長（単位：バイト） |
|---|---|
| 0 ～ 1012 | 4096 |
| 1013 ～ 2036 | 8192 |
| 2037 ～ 4084 | 16384 |
| 4085 ～ 16360 | 32768 |
| 16361 ～ 32720 | ↑（ハッシュ表最大行長＋ 48）÷ 2048 ↑× 2048 |

c：ハッシュ表最大行長
ハッシュ表最大行長については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

### （4）スナップショット方式指定時に必要なメモリ所要量の求め方

pd_pageaccess_mode オペランドを省略した場合又は SNAPSHOT を指定した場合，スナップショット方式を適用する SQL 文を実行すると，データベース検索時のページアクセス方式にスナップショット方式を使用します。このとき，表又はインデクスの格納 RD エリアのページサイズに基づいて，動的に次に示すメモリサイズのプロセス固有領域を確保します。

**計算式**

a × 2　（単位：バイト）

a：検索対象の表又はインデクスが格納されている RD エリア中の最大ページ長
ただし，LOB 用 RD エリアは除きます。

### （5）先頭から n 行の検索結果を取得する機能実行時に必要なメモリ所要量の求め方

先頭から n 行の検索結果を取得する機能を使用すると，検索結果の先頭（又はユーザが指定した先頭からのオフセット行数分読み飛ばした位置）から n 行取得できます。

LIMIT 句に指定した行数が 1 以上で，（オフセット行数＋ LIMIT 句に指定した行数）の値が 32,767 以下の場合，（オフセット行数＋ LIMIT 句に指定した行数）以内に入り得る行をメモリに保持します。確保す

るプロセス固有領域のメモリサイズは，次に示す計算式で求められます。なお，（オフセット行数＋LIMIT 句に指定した行数）の値が 32,768 以上になる場合は作業表を作成するため，「18. 作業表用ファイルの容量の見積もり」を参照してください。

**計算式**

{100 ＋（a ＋ 2）×（オフセット行数＋ LIMIT 句に指定した行数）} × b
（単位：バイト）

a：行長
行長は 32,720 バイト以下でなければなりません。行長は次の計算式で求められます。

$$\sum_{i=1}^{m}(Ai) + 2 \times m + 4 + c$$
（単位：バイト）

m：選択式，GROUP BY 句，又は ORDER BY 句に指定した列数
FOR UPDATE 句を指定した場合は 1 を加算してください。ただし，選択式に ROW を指定している場合は表の全列数になります。

Ai：先頭 n 行保持領域に格納する行の i 番目の列データ長
列のデータ長については，「表 16-1　データ長一覧」を参照し，d に定義長を代入して求めてください。
ただし，BLOB データ，定義長が 256 バイト以上の文字データ（各国・混在文字データも含む），BINARY データのうち，下記に属さない列の場合は 12 バイトになります。
- DISTINCT 句指定の選択式に指定する列
- UNION [ALL] によって集合演算対象となっている問合せ指定中の選択式
- ORDER BY 句に指定した列

また，FOR UPDATE 句を指定した場合に，m に加算した 1 に対応する Ai は 12 バイトとします。

c：8
ただし，次の場合は 0 になります。
- 検索対象の表に EX モードで排他が掛かっている場合
- WITHOUT LOCK を指定した場合
- グループ分け高速化機能を指定した場合
- 複数の表を結合する場合

b：先頭 n 行保持領域数
先頭 n 行の保持領域数は次の計算式で求められます。

1 ＋ UNION [ALL] 句指定数

## （6）探索条件にインデクス型プラグイン専用関数を指定した SQL 文実行時に必要なメモリ所要量の求め方

探索条件にインデクス型プラグイン専用関数を指定した SQL 文の実行時に確保するプロセス固有領域のメモリサイズは，次に示す計算式で求められます。

**計算式**

a × 500 ＋（20 ＋ 6）× 800 ＋ 16　（単位：バイト）

a：行長。行長は次の計算式で求められます。

$$\sum_{i=1}^{m}(Ai) + 4 \times (m + 2) + 12 + 4 + 8$$

（単位：バイト）

m：選択式，結合条件，GROUP BY 句，又は ORDER BY 句に指定した列数
FOR UPDATE 句を指定した場合は 1 を加算してください。ただし，選択式に ROW を指定している場合は表の全列数になります。

Ai：取り出す行の i 番目の列データ長
列のデータ長については，「表 16-1　データ長一覧」を参照し，d に定義長を代入して求めてください。
ただし，BLOB データ，又は定義長が 256 バイト以上の文字データ（各国・混在文字データも含む）で，下記に属さない列の場合は 12 バイトになります。

- 結合条件中に指定する列（結合列）
- DISTINCT 句指定の選択式に指定する列
- 限定述語の副問合せ中の選択式に指定する列
- IN 述語の副問合せ中の選択式に指定する列
- UNION [ALL]，又は EXCEPT [ALL] によって集合演算対象となっている問合せ指定中の選択式
- ORDER BY 句に指定した列

また，FOR UPDATE 句を指定した場合に，m に加算した 1 に対応する Ai は 12 バイトとします。

## （7）拡張 SQL エラー情報出力機能使用時に必要なメモリ所要量の求め方

拡張 SQL エラー情報出力機能を使用した場合，次のときにプロセス固有領域を確保します。

### （a）OPEN 文実行時

**計算式**

**● 32 ビットモードの場合**
（16 ＋ 16 × m）＋ a
**● 64 ビットモードの場合**
（16 ＋ 24 × m）＋ a

（単位：バイト）

a：？パラメタ又は埋込み変数のデータ長の合計

$$a = \sum_{i=1}^{m}(ai)$$

m：SQL 文中の？パラメタ又は埋込み変数の数

ai：i 番目の？パラメタ又は埋込み変数のデータ長

埋込み変数又は？パラメタのデータ長を次の表に示します。

**表 15-5 埋込み変数又は？パラメタのデータ長**

| データ型 | 列長（標識変数なし） | 列長（標識変数あり，埋込み変数又は？パラメタ） |
|---|---|---|
| INTEGER | 4 | 6 |
| SMALLINT | 2 | 4 |
| DECIMAL(p，s) | ↑（p＋1）÷2↑ | ↑（p＋5）÷2↑ |
| FLOAT | 8 | 10 |
| SMALLFLT | 4 | 6 |
| INTERVAL YEAR TO DAY | 5 | 7 |
| INTERVAL HOUR TO SECOND | 4 | 6 |
| CHAR(n) | n | n＋2 |
| VARCHAR(n) | n＋2 | n＋4 |
| NCHAR(n) | 2×n | 2×n＋2 |
| NVARCHAR(n) | 2×n＋2 | 2×n＋4 |
| MCHAR(n) | n | n＋2 |
| MVARCHAR(n) | n＋2 | n＋4 |
| DATE | 4 | 6 |
| TIME | 3 | 5 |
| BLOB(n) | n＋4 | n＋8 |
| TIMESTAMP(p) | 7＋（p÷2） | 9＋（p÷2） |
| BINARY(n) | n＋4 | n＋8 |

#### （b）定義系 SQL の PREPARE 文実行時

**計算式**

SQL 文長＋20

（単位：バイト）

### （8）部分構造インデクスの定義，又は部分構造インデクスを定義した表の更新時に必要なメモリ所要量の求め方

#### （a）部分構造インデクスの定義時

定義系 SQL の CREATE INDEX で部分構造インデクスを定義する場合に確保するプロセス固有領域は，次に示す計算式で求められます。

**計算式**

（インデクスキー長※×100＋64）　（単位：バイト）

注※

表に定義する部分構造インデクスの最大定義長です。

#### (b) 部分構造インデクスを定義した表の更新時

操作系 SQL の INSERT，UPDATE，又は DELETE で部分構造インデクスを定義した表を更新する場合に確保するプロセス固有領域は，次に示す計算式で求められます。

**計算式**

（インデクスキー長※1 × 100 ＋ 64 ＋ 128）＋ Σ（インデクスキー長＋ 128）※2　（単位：バイト）

注※ 1

表に定義している部分構造インデクスの最大定義長です。

注※ 2

USING UNIQUE TAG 指定の部分構造インデクス数です。

## 15.1.7 SQL 前処理時に必要なメモリ所要量の計算式

### (1) ストアドプロシジャを使用しない場合に必要なメモリ所要量の求め方

ストアドプロシジャを使用しない場合，SQL 前処理時に確保するメモリサイズは，次に示す計算式で求められます。

**計算式**

```
↑ {
    2586 ＋ Si × 60 ＋ Pi × 20 ＋ Ti × 1424 ＋ Ci × Ti × 72 ＋ Wi × 776 ＋ Ti × Wi × 72
    ＋ Ki × 276 ＋ Ki × Ti × 72 ＋ Li × 3 ＋ Li × Ti ＋ Di × Ti × 134 ＋ Ari × 108
    ＋ Gi × 44 ＋ Sli × 40 ＋ Upi × 110 ＋ Fi × 90 ＋ Ti × Cwi × 48
    ＋ MAX（Pi，Wpi）× 60
  } × 1.2 ÷ 1024 ↑
                                                    （単位：キロバイト）
```

Si：SQL 文中の検索項目数

Pi：SQL 文中の埋込み変数，？パラメタ又は SQL パラメタの数

Ti：SQL 文中の表名の数

Ci：SQL 文中の列名の数

Wi：SQL 文中の論理演算子（AND 及び OR）に出てくる述語の数

Ki：SQL 文中の定数の数

Li：SQL 文中の定数の長さの合計（単位：バイト）

Di：SQL 文中に定義された格納条件の総数

Ari：SQL 文中の四則演算及び連結演算の数

Gi：SQL 文中の GROUP BY 句に指定した列の数

Ori：SQL 文中の ORDER BY 句に指定した列指定又はソート項目指定番号の数

Fi：SQL 文中の集合関数及びスカラ関数の総数

Sli：SQL 文中の問合せ指定の数

Upi：SQL 文中の更新列数

Cwi：SQL 文中の CASE 式中の WHEN の数

Wpi：SQL 文中の WITH 句に対応する変数の数

注

SELECT_APSL が適用されている場合は，前記の計算式で求めた値を 3 倍してください。
SELECT_APSL が適用されているかどうかについては，アクセスパス表示ユティリティ（pdvwopt）を使用すると分かります。アクセスパス表示ユティリティ（pdvwopt）については，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

### (2) ストアドプロシジャを使用する場合に必要なメモリ所要量の求め方

ストアドプロシジャを使用する場合，SQL 前処理時に確保するメモリサイズ（単位：キロバイト）は，「(1) ストアドプロシジャを使用しない場合に必要なメモリ所要量の求め方」の計算式で求めた値に，ストアドプロシジャごとのプロシジャ制御用オブジェクト長を加算します。プロシジャ制御用オブジェクト長の計算式については，システム共通定義の pd_sql_object_cache_size オペランドの 1 ストアドプロシジャのプロシジャ制御用オブジェクト長を参照してください。1 ストアドプロシジャのプロシジャ制御用オブジェクト長については，マニュアル「HiRDB Version 8 システム定義」の「1 ルーチンのルーチン制御用オブジェクト長の計算式」を参照してください。

## 15.1.8 BLOB 型データの検索又は更新時に必要なメモリ所要量の計算式（HiRDB/ シングルサーバの場合）

BLOB 型データの検索又は更新時に必要なメモリ所要量は次に示す計算式で求められます。

**計算式**

```
a ＋ b ＋ 17　　（単位：キロバイト）
```

a：1SQL 文中に指定する BLOB 型入力変数又は出力変数で，実行する SQL 文の中で次に示す計算式の結果が最大となる値です。

```
↑ {
     c
    Σ（BLOB 型入力変数 i の実長※1 ＋ 118）＋
   i=1
     d
    Σ（BLOB 型出力変数 j の定義長※2 ＋ 86）
   j=1
} ÷ 1024 ↑
```

注※1

埋込み変数で UAP から HiRDB サーバに受け渡された BLOB 型データの実際の長さです。

注※2

HiRDB から UAP に返す BLOB 型データを受け取る UAP の埋込み変数の宣言長です。INSERT － SELECT 文の場合は，SELECT 側で射影する BLOB 列を出力変数とみなします。

b：同時オープン中のカーソルで結合検索を行う SQL 文の組み合わせで，次に示す計算式の結果が最大となる値です。

```
↑ {
     e
     Σ {
    i=1
     d
     Σ（BLOB 型出力変数 j の定義長＋ 18)}
    j=1
} ÷ 1024 ↑
```

c：入力変数の数

d：出力変数の数

e：同時オープン中のカーソル数

## 15.1.9 ブロック転送又は配列 FETCH で必要なメモリ所要量の計算式

ブロック転送又は配列 FETCH で必要なメモリ所要量は，次の計算式で求められます。

| 条件 | | PDBLKBUFFSIZE オペランドの指定値 | |
|---|---|---|---|
| | | 省略又は 0 | 1 以上 |
| FETCH 文の INTO 句に配列型の埋込み変数を指定する | | 計算式 1 | |
| FETCH 文の INTO 句に配列型の埋込み変数を指定しない | PDBLKF オペランドを省略又は 1 | － | 計算式 2 |
| | PDBLKF オペランドが 2 以上 | 計算式 1 | |

（凡例）－：該当しません。

**計算式 1**

```
↑ {864 ＋ 16 × a ＋（6 × a ＋ 2 × d ＋ b）× c} ÷ 1024 ↑
                                        （単位：キロバイト）
```

a：SELECT 句で指定する検索項目数

b：FETCH 文で受け取る検索結果中の 1 行のデータ長（各列の最大長の合計。単位はバイト）

c：PDBLKF オペランドの指定値又は配列数

d：SELECT 句で指定する検索項目で，BINARY 型を指定した選択式の数

**計算式 2**

```
MAX（X1，X2）
                                        （単位：キロバイト）
```

$X_1$：↑（864 ＋ 22 × a ＋ 2 × c ＋ b）÷ 1024 ↑

$X_2$：PDBLKBUFFSIZE オペランドの値

a：SELECT 句で指定する検索項目数

b：FETCH 文で受け取る検索結果中の 1 行のデータ長（実際に取得する各列の長さの合計。単位はバイト）

c：SELECT 句で指定する検索項目で，BINARY 型を指定した選択式の数

## 15.1.10　インメモリデータ処理で必要なメモリ所要量

インメモリデータ処理で必要なメモリ所要量は次に示す計算式で求められます。

**計算式**

**●インメモリデータバッファが使用する共用メモリを実メモリ上に固定しない場合**
計算式 1 ＋ D × 2　（単位：キロバイト）
**●インメモリデータバッファが使用する共用メモリを実メモリ上に固定する場合**
計算式 1 ＋ D ×↑（↑ 2048 ÷ p ↑× p）÷ 1024 ↑　（単位：キロバイト）

**計算式 1**

**●インメモリデータバッファが使用する共用メモリを実メモリ上に固定しない場合**

$$\sum_{i=1}^{n} \uparrow \{704+32\times A+48+448\times B+2048+C\times B\} \div 1024 \uparrow$$

（単位：キロバイト）

**●インメモリデータバッファが使用する共用メモリを実メモリ上に固定する場合**

$$\sum_{i=1}^{n} \uparrow \{\uparrow (704+32\times A+48+448\times B+2048+C\times B) \div p \uparrow \times p\} \div 1024 \uparrow$$

（単位：キロバイト）

n：インメモリ RD エリアの数

A：インメモリ RD エリアを構成する HiRDB ファイル数

B：インメモリ RD エリアの総ページ数

C：インメモリ RD エリアのページサイズ

D：計算式 2 の値

p：Windows の Large Page でのページサイズ

pdntenv コマンドで確認できます。

**計算式 2（インメモリデータバッファが使用する共用メモリセグメント数）**

↑計算式 1 の値÷（SHMMAX オペランドの値× 1024）↑

計算式 2 は 1RD エリア当たりの計算式です。インメモリ RD エリアの数だけ計算してください。

計算式 2 で求めた値は，pd_max_resident_rdarea_shm_no オペランドの見積もりに使います。

# 15.2 HiRDB/ パラレルサーバのメモリ所要量の見積もり

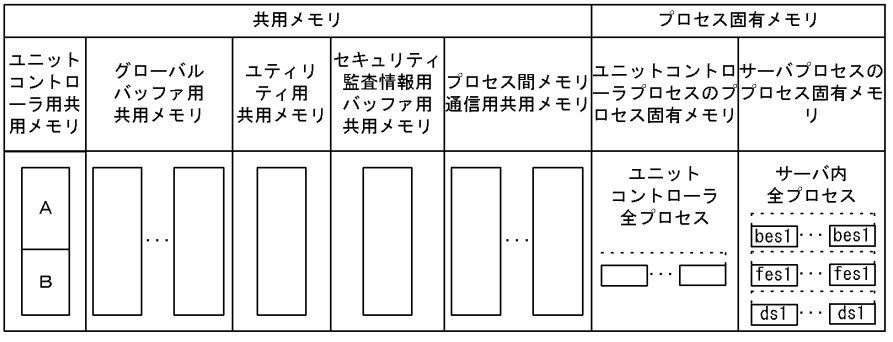
ここでは，HiRDB/ パラレルサーバを構成する各ユニットのメモリ所要量の見積もり方法について説明します。ここで説明する項目を次に示します。

- メモリ配置
- メモリ所要量の計算式
- ユニットコントローラが使用する共用メモリの計算式
- 各サーバが使用する共用メモリの計算式
- グローバルバッファが使用する共用メモリの計算式
- SQL 実行時に必要なメモリ所要量の計算式
- SQL 前処理時に必要なメモリ所要量の計算式
- BLOB 型データの検索又は更新時に必要なメモリ所要量の計算式（フロントエンドサーバの場合）
- ブロック転送又は配列 FETCH で必要なメモリ所要量の計算式（フロントエンドサーバの場合）
- BLOB 型データの検索又は更新時に必要なメモリ所要量の計算式（バックエンドサーバ又はディクショナリサーバの場合）

## 15.2.1 メモリ配置

HiRDB/ パラレルサーバの各ユニットのメモリ配置を次の図に示します。

**図 15-2 HiRDB/ パラレルサーバの各ユニットのメモリ配置**




（凡例）A：ユニットコントローラの各プロセス使用分
B：各サーバのプロセス使用分
bes：バックエンドサーバ fes：フロントエンドサーバ ds：ディクショナリサーバ

HiRDB/ パラレルサーバの各ユニットの共用メモリの詳細を次の表に示します。

**表 15-6 HiRDB/ パラレルサーバの各ユニットの共用メモリの詳細**

| 項目 | 共用メモリの種類 | | | | |
|---|---|---|---|---|---|
| | ユニットコントローラ用共用メモリ | グローバルバッファ用共用メモリ | ユティリティ用共用メモリ | セキュリティ監査情報用バッファ用共用メモリ | プロセス間メモリ通信用共用メモリ |
| 使用目的 | システム制御 | グローバルバッファ | ユニットコントローラとユティリティとの通信 | セキュリティ監査情報用バッファ | クライアント－サーバプロセス間メモリ通信 |

| 項目 | 共用メモリの種類 | | | | |
|---|---|---|---|---|---|
| | ユニットコントローラ用共用メモリ | グローバルバッファ用共用メモリ | ユティリティ用共用メモリ | セキュリティ監査情報用バッファ用共用メモリ | プロセス間メモリ通信用共用メモリ |
| 使用プロセス | 全 HiRDB プロセス | バックエンドサーバ，ディクショナリサーバ | ユティリティプロセス | フロントエンドサーバ | フロントエンドサーバ，クライアントプロセス |
| セグメント数 | 1 個 | • グローバルバッファの動的変更機能を使用しない場合<br>1 ～ 16 個※1<br>• グローバルバッファの動的変更機能を使用している場合<br>32 ビットモード：<br>1 ～ 516 個※1<br>64 ビットモード<br>：1 ～ 1,016 個※1 | 1 個 | 1 個 | 環境変数 PDIPC=MEMORY で接続中のクライアント数（0 ～ 2000）× 2 個 |
| 1 セグメントの上限 | 「表 15-7 HiRDB/ パラレルサーバの各ユニットが使用するメモリ所要量」を参照してください。ただし，OS の環境によっては確保できない場合があります。 | SHMMAX オペランドの値でセグメントを分割します。ただし，OS の環境によっては確保できない場合があります。 | 「表 15-7 HiRDB/ パラレルサーバの各ユニットが使用するメモリ所要量」を参照してください。ただし，OS の環境によっては確保できない場合があります。 | 「表 15-7 HiRDB/ パラレルサーバの各ユニットが使用するメモリ所要量」を参照してください。ただし，OS の環境によっては確保できない場合があります。 | 「表 15-7 HiRDB/ パラレルサーバの各ユニットが使用するメモリ所要量」を参照してください。ただし，OS の環境によっては確保できない場合があります。 |
| 確保条件 | なし | グローバルバッファ定義が存在すること | pd_utl_exec_mode=1 指定 | pd_aud_file_name オペランドによる監査証跡ファイル用の HiRDB ファイルシステム領域名の指定 | クライアント環境変数 PDIPC=MEMORY で接続中クライアントあり |
| 作成契機 | ユニット起動時（ユーザサーバホットスタンバイまたは高速系切り替え機能使用時の待機ユニットの起動を含む） | • サーバ起動時（高速系切り替え機能使用時の待機ユニットの起動を含む）<br>• pdbufmod -k {add\|upd} 実行時 | ユティリティ実行時 | フロントエンドサーバ起動時 | クライアントとサーバの接続時 |

| 項目 | 共用メモリの種類 | | | | |
|---|---|---|---|---|---|
| | ユニットコントローラ用共用メモリ | グローバルバッファ用共用メモリ | ユティリティ用共用メモリ | セキュリティ監査情報用バッファ用共用メモリ | プロセス間メモリ通信用共用メモリ |
| 削除契機 | ユニット起動時（ユーザサーバホットスタンバイまたは高速系切り替え機能使用時の待機ユニットの起動を含む） | • pdbufmod -k del 実行時<br>• 正常終了又は計画停止の場合：サーバ終了時<br>• 強制停止，異常終了，又は高速系切り替え機能使用時の待機ユニットの停止の場合：次回ユニット起動時 | ユティリティ終了後 10 分後 | フロントエンドサーバ終了時 | クライアントとサーバの接続切り離し時 |
| pdls -d mem による表示 | 表示される | 表示される | 表示される | 表示される | 表示されない |
| pdls -d mem の SHM-OWNER | MANAGER | サーバ名 | UTILITY | AUDDEF | 表示なし |
| 関連オペランド | • pd_dic_shmpool_size<br>• pd_bes_shmpool_size | • pd_dbbuff_modify<br>• pdbuffer<br>• SHMMAX | • pd_utl_exec_mode | • セキュリティ監査機能に関するオペランド※2 | • PDIPC<br>• PDSENDMEMSIZE<br>• PDRECVMEMSIZE |
| 備考 | — | — | pd_utl_exec_mode=1 の場合だけ作成されます（pd_utl_exec_mode=0 の場合，該当する領域はユニットコントローラ用共用メモリ内に確保されます）。 | — | — |

（凡例）—：該当しません。

注※ 1
グローバルバッファ割り当てバックエンドサーバ又はディクショナリサーバ当たりの数です。

注※ 2
詳細はマニュアル「HiRDB Version 8 システム定義」を参照してください。

## 15.2.2 メモリ所要量の計算式

HiRDB/ パラレルサーバの各ユニットが使用するメモリ所要量は，次の表に示すすべての項目を加算した値です。

**表 15-7 HiRDB/ パラレルサーバの各ユニットが使用するメモリ所要量**

| 項目 | | | | メモリ所要量（単位：キロバイト） |
|---|---|---|---|---|
| プロセス固有領域 | ユニットコントローラ全プロセスが使用するプロセス固有領域 | | | **● 32 ビットモードの場合**<br>J ＋ K ×ユニット内 FES 数＋ L ×（ユニット内 BES 数＋ユニット内 DS 数）＋↑ {(64 ＋ 48 ×（v ＋ 1)）×（pd_max_server_process の値− w）＋ z} ÷ 1024 ↑<br>**● 64 ビットモードの場合**<br>J ＋ K ×ユニット内 FES 数＋ L ×（ユニット内 BES 数＋ユニット内 DS 数）＋↑ {(64 ＋ 64 ×（v ＋ 1)）×（pd_max_server_process の値− w）＋ z} ÷ 1024 ↑<br><br>**●プラグインを使用する場合，加算します。**<br>＋ 1400<br>**●非同期 READ 機能を使用する場合，加算します。**<br>＋ s<br>**● pd_process_terminator オペランドに fixed を指定した場合，加算します。**<br>＋ M ×（pd_process_terminator_max の値− 1）<br>**●インメモリデータ処理を行う場合，加算します。**<br>＋↑ {T ×（pd_max_bes_process の値× 2 ＋ 7）×ユニット内 BES 数} ÷ 1024 ↑<br>**●通信トレース格納最大数を変更する場合，加算します。**<br>＋↑ V ÷ 1024 ↑ |
| | 各サーバプロセスが使用するプロセス固有領域※1 ※2 | フロントエンドサーバ | | (N ＋ h ＋ m ＋ p ＋ q）×（b ＋ 3）＋ 100 ＋ y<br>**●通信トレース格納最大数を変更する場合，加算します。**<br>＋↑ W ÷ 1024 ↑ |
| | | ディクショナリサーバ | pd_work_buff_mode= each 指定時 | {(P ＋ i ＋ m ＋ r ＋ t）×（b ＋ 3)} ＋ (a ＋ 9）× 2 ＋ 100 ＋ y ＋ S<br>**●通信トレース格納最大数を変更する場合，加算します。**<br>＋↑ W ÷ 1024 ↑ |
| | | | pd_work_buff_mode= pool 指定又は省略時 | {(P ＋ i ＋ m ＋ r ＋ t）×（b ＋ 3)}<br>＋ a ＋ 9 ＋↑ a ÷ 128 × 0.1 ↑＋ 100 ＋ n ＋ y ＋ S<br>**●通信トレース格納最大数を変更する場合，加算します。**<br>＋↑ W ÷ 1024 ↑ |
| | | バックエンドサーバ | pd_work_buff_mode= each 指定時 | {Q ＋ g ＋（a ＋ 9）× c ＋ i ＋ m ＋ r ＋ t}<br>×（b ＋ 3）＋ 100 ＋ y ＋ S<br>**●インメモリデータ処理を行う場合，加算します。**<br>＋↑ {T ×（pd_max_users の値＋ 3)} ÷ 1024 ↑<br>**●通信トレース格納最大数を変更する場合，加算します。**<br>＋↑ W ÷ 1024 ↑ |

| 項目 | | | | メモリ所要量（単位：キロバイト） |
|---|---|---|---|---|
| | | | pd_work_buff_mode=pool 指定又は省略時 | ●32 ビットモードの場合<br>（Q ＋ g ＋ a ＋ 9<br>＋↑ a ÷ 128 × 0.1 ＋ 11 ↑＋ i ＋ m ＋ r ＋ t）<br>×（b ＋ 3）＋ 100 ＋ n ＋ y ＋ S<br>●64 ビットモードの場合<br>（Q ＋ g ＋ a ＋ 9<br>＋↑ a ÷ 128 × 0.1 ＋ 15 ↑＋ i ＋ r ＋ t）<br>×（b ＋ 3）＋ 100 ＋ n ＋ y ＋ S<br>●インメモリデータ処理を行う場合，加算します。<br>＋↑ {T ×（pd_max_users の値＋ 3)} ÷ 1024 ↑<br>●通信トレース格納最大数を変更する場合，加算します。<br>＋↑ W ÷ 1024 ↑ |
| 共用メモリ | ユニットコントローラ用共用メモリ中，ユニットコントローラが使用する領域 | | | ↑ d ÷ 1024 ↑ |
| | ユニットコントローラ用共用メモリ中，各サーバが使用する領域※1 | | | e |
| | グローバルバッファ用共用メモリ | | | f |
| | インメモリデータ処理用共用メモリ | | | U |
| | ユティリティ用共用メモリ | | | u |
| | セキュリティ監査情報用バッファ用共用メモリ | | | ●システムによる自動計算の場合<br>↑ 0.3 ＋ MAX {(R ＋ 100)，(R × 1.2)} × 0.25 ↑<br>●ユーザ指定値（pd_audit_def_buffer_size オペランドを指定）にする場合<br>pd_audit_def_buffer_size の指定値 |
| | プロセス間メモリ通信用共用メモリ※3 | | | j × k |

注※1

ユニット内に複数のサーバ（システムマネジャを除きます）がある場合は，サーバごとに計算してください。

注※2

プラグインを使用する場合は，1 サーバプロセス当たり 300 を加算してください。

注※3

クライアント環境定義で PDIPC=MEMORY を指定した場合に加算します。プロセス間メモリ通信機能及びクライアント環境定義については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

なお，HiRDB サーバ又は HiRDB クライアントのどちらかが 32 ビットモードの場合，プロセス間メモリ通信機能で使用する共用メモリは 32 ビット空間内に確保されます。

a：pd_work_buff_size オペランドの値

b：pd_max_users オペランドの値

- ディクショナリサーバの場合は，pd_max_dic_process オペランドの値となります。
- バックエンドサーバの場合は，pd_max_bes_process オペランドの値となります。
- pd_max_dic_process 又は pd_max_bes_process オペランドを省略する場合は，pd_max_users オペランドの値となります。

c：最大作業表数

SQL 文ごとの作業表数を「表 15-8　SQL 文ごとの作業表数の求め方」から求めます。表 15-8 から求めた作業表数のうちで最大のものを最大作業表数とします。

d：「15.2.3　ユニットコントローラが使用する共用メモリの計算式」で求めた値

e：「15.2.4　各サーバが使用する共用メモリの計算式」で求めた値

f：「15.2.5　グローバルバッファが使用する共用メモリの計算式」で求めた値

g：SQL 実行時に必要なメモリ所要量
計算式については，「15.2.6　SQL 実行時に必要なメモリ所要量の計算式」を参照してください。

h：SQL 前処理時に必要なメモリ所要量
計算式については，「15.2.7　SQL 前処理時に必要なメモリ所要量の計算式」を参照してください。

i：LOB バッファ一括入出力ワークメモリ
該当するサーバの LOB 用 RD エリアに LOB 用グローバルバッファを指定している場合だけ（システム共通定義の pdbuffer オペランドに -b を指定している場合），62 キロバイトを加算してください。

j：プロセス間メモリ通信機能を使用するクライアントの最大同時実行数
分からない場合は，プロセス間メモリ通信機能を使用する全クライアント数，又は pd_max_users オペランドの値を代入してください。

k：プロセス間メモリ通信機能を使用する全クライアントのデータ送受信用メモリサイズ（クライアント環境定義の PDSENDMEMSIZE の値＋ PDRECVMEMSIZE の値）の平均値

m：Java 仮想マシンが使用するメモリ所要量
Java ストアドプロシジャ又は Java ストアドファンクションを使用する場合に，Java 仮想マシンが使用するメモリ所要量を加算します。Java 仮想マシンが使用するメモリ所要量は，Java 仮想マシンのオプションや Java 仮想マシンのバージョンによって異なります。Java 仮想マシンが使用するメモリ所要量については，Java 仮想マシンのマニュアルを参照してください。

n：作業表用増分メモリサイズ
pd_work_buff_expand_limit オペランドを指定する場合に作業表用増分メモリサイズを加算します。作業表用増分メモリサイズは次に示す計算式から求めます。
**作業表用増分メモリサイズ（キロバイト）＝作業表用増分バッファサイズ＋↑（作業表用の増分バッファサイズ÷ 128）× 0.1 ↑**
- 作業表用増分バッファサイズ（キロバイト）＝ MAX（0，ハッシュジョイン，副問合せのハッシュ実行による作業表用増分バッファサイズ）＋ MAX（0，作業表数の増加による作業表用増分バッファサイズ）
- ハッシュジョイン，副問合せのハッシュ実行による作業表用増分バッファサイズ＝ MIN｛(ハッシュジョイン，副問合せのハッシュ実行をするときの作業表用バッファサイズ－ pd_work_buff_size オペランドの値)，(pd_work_buff_expand_limit オペランドの値－ pd_work_buff_size オペランドの値)｝×ハッシュジョイン，副問合せのハッシュ実行をする同時実行ユーザ数
ハッシュジョイン，副問合せのハッシュ実行をするときの作業表用バッファサイズの求め方については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。
- 作業表数の増加による作業表用増分バッファサイズ＝ MIN｛(使用作業表数× 128 － pd_work_buff_size オペランドの値)，(pd_work_buff_expand_limit オペランドの値－ pd_work_buff_size オペランドの値)｝×（使用作業表数が pd_work_buff_size オペランドの値÷ 128 以上になるユーザ数）

使用作業表数＝ MAX（1SQL 文が使用する作業表用ファイルの数，ASSIGN LIST 文が使用する作業表用ファイルの数）
1SQL 文が使用する作業表用ファイルの数，及び ASSIGN LIST 文が使用する作業表用ファイルの数の求め方については，「18.3　最大ファイル数の見積もり（pdfmkfs -l コマンド）」を参照してください。

p：BLOB 型データ用に必要なメモリ所要量
計算式については，「15.2.8　BLOB 型データの検索又は更新時に必要なメモリ所要量の計算式（フロントエンドサーバの場合）」を参照してください。

q：サーバ側でブロック転送又は配列 FETCH で必要なメモリ所要量
計算式については，「15.2.10　ブロック転送又は配列 FETCH で必要なメモリ所要量の計算式（フロントエンドサーバの場合）」を参照してください。

r：BLOB 型データ用に必要なメモリ所要量
計算式については，「15.2.9　BLOB 型データの検索又は更新時に必要なメモリ所要量の計算式（バックエンドサーバ又はディクショナリサーバの場合）」を参照してください。

s：非同期 READ 用メモリサイズ
非同期 READ 機能を使用する場合に加算します。次に示す計算式から求めます（単位：キロバイト）。

```
（90 ＋
90
Σ RD エリア用 HiRDB ファイルシステム領域管理用メモリ）
i=1
× pd_max_ard_process の値
```

RD エリア用 HiRDB ファイルシステム領域管理用メモリは計算値の大きい順に 90 領域を計算に使用します。サーバで使用する領域数が 90 領域に満たない場合，その領域まで計算します。
HiRDB ファイルシステム領域管理用メモリはそれぞれの領域の初期設定時のパラメタを使用して，次の計算式から求めます（単位：キロバイト）。

```
{（ファイル数※1＋増分数※2）÷ 64} × 1.5※3
```

注※ 1　pdfmkfs -l 指定値です。
注※ 2　pdfmkfs -e 指定値です。
注※ 3　領域サイズ（pdfmkfs -n 指定値）が 2048 以上の場合に乗算します。

t：HiRDB ファイルシステム用メモリサイズ
次に示す計算式から求めます（単位：キロバイト）。

```
347 ＋作業表用 HiRDB ファイルシステム領域管理用メモリ＋
システムログ用 HiRDB ファイルシステム領域管理用メモリ＋
90
ΣRD エリア用 HiRDB ファイルシステム領域管理用メモリ
i=1
```

作業表用及びシステムログ用 HiRDB ファイルシステム領域管理用メモリは，サーバで使用する HiRDB ファイルシステム領域で計算値が最大になるものを使用します。RD エリアの場合は計算値の大きい順に 90 領域を計算に使用します。サーバで使用する領域数が 90 領域に満たない場合，その領域まで計算します。
HiRDB ファイルシステム領域管理用メモリはそれぞれの領域の初期設定時のパラメタを使用して，次の計算式から求めます（単位：キロバイト）。

{(ファイル数※1＋増分数※2）÷ 64} × 1.5※3

注※ 1　pdfmkfs -l 指定値です。
注※ 2　pdfmkfs -e 指定値です。
注※ 3　領域サイズ（pdfmkfs -n 指定値）が 2048 以上の場合に乗算します。

u：pd_utl_exec_mode の値が 0 の場合：0
pd_utl_exec_mode の値が 1 の場合：↑ {(b × 2000 ＋ 136）÷ 1024} ↑× 1024

v：ユニット制御情報定義として有効な pd_module_trace_max の値

w：ユニット内の全サーバプロセスに対して，（最大起動プロセス数＋ 3）を合計した値
最大起動プロセス数については，マニュアル「HiRDB Version 8 システム定義」を参照してください。

x：ユニット制御情報定義として有効な pd_pth_trace_max 値です。オペランド指定値を 2 のべき乗の値に切り上げた値になります。

y：ユニット内の各サーバプロセスに対して次の計算式で求めた値の総和
32 ビットモードの場合：
↑ {(64 ＋ 48 ×（pd_module_trace_max の値＋ 1)）×（最大起動プロセス数＋ 3)} ÷ 1024 ↑
64 ビットモードの場合：
↑ {(64 ＋ 64 ×（pd_module_trace_max の値＋ 1)）×（最大起動プロセス数＋ 3)} ÷ 1024 ↑
最大起動プロセス数については，マニュアル「HiRDB Version 8 システム定義」を参照してください。

z：HiRDB の再開始用メモリサイズ
このメモリサイズを確保できないと，HiRDB の再開始に失敗します。次の計算式から求めます（単位：バイト)。

（D ＋ E ＋ F）×ディクショナリサーバ数＋（D ＋ E ＋ F）×バックエンドサーバ数＋ ΣH

HiRDB の再開始用メモリサイズを求める計算に使用する変数を次に示します。

| 変数 | 値 |
|---|---|
| D | **● 32 ビットモードの場合**<br>246762 ＋ 4 × pd_max_rdarea_no の値<br>＋ {48 ×（pd_max_rdarea_no の値＋表数）＋ 304} ×（pd_max_users の値※× 2 ＋ 7）<br>**● 64 ビットモードの場合**<br>305274 ＋ 8 × pd_max_rdarea_no の値<br>＋ {64 ×（pd_max_rdarea_no の値＋表数）＋ 512} ×（pd_max_users の値※× 2 ＋ 7）<br><br>**表数：**62 ＋ MAX {pd_max_access_tables の値，500} |

| 変数 | 値 |
|---|---|
| E | b1 × X ＋ b2 × Y<br><br>**b1：サーバ用ステータスファイルのレコード長＜** 4096 **の場合**<br>MAX（(↓（3400 ÷（(↓（(レコード長－ 40）－ 308）÷ 20 ↓）<br>＋（↓（レコード長－ 40）÷ 20 ↓）×（MAX（↓ 4096 ÷レコード長↓，2）－ 1））<br>＋ 0.7）↓），1）× MAX(↓ 4096 ÷レコード長↓，2）×（レコード長－ 40）<br>4096 **≦サーバ用ステータスファイルのレコード長＜** 12288 **の場合**<br>MAX（↓（3400 ÷（↓（(((レコード長－ 40）－ 308）÷ 20）↓）＋ 0.7）↓，1）<br>×（レコード長－ 40）<br>12288 **≦サーバ用ステータスファイルのレコード長の場合**<br>MAX（↓（3400 ÷（↓（(((レコード長－ 40）－ 836）÷ 20）↓）＋ 0.7）↓，1）<br>×（レコード長－ 40）<br>**X：**サーバ内の RD エリア数≦ 3400 の場合：1<br>3401 ≦サーバ内の RD エリア数≦ 6800 の場合：2<br>6801 ≦サーバ内の RD エリア数の場合：3<br>b2**：サーバ用ステータスファイルのレコード長＜** 4096 **の場合**<br>（↓（5662310 ÷（(↓（(レコード長－ 40）－ 308）÷ 20 ↓）<br>＋（↓（レコード長－ 40）÷ 20 ↓）×（MAX（↓ 4096 ÷レコード長↓，2）－ 1））<br>＋ 0.7）↓）× MAX（↓ 4096 ÷レコード長↓，2）×（レコード長－ 40）<br>4096 **≦サーバ用ステータスファイルのレコード長＜** 12288 **の場合**<br>↓（5662310 ÷（↓（(((レコード長－ 40）－ 308）÷ 20）↓）＋ 0.7）↓<br>×（レコード長－ 40）<br>12288 **≦サーバ用ステータスファイルのレコード長の場合**<br>↓（5662310 ÷（↓（(((レコード長－ 40）－ 836）÷ 20）↓）＋ 0.7）↓<br>×（レコード長－ 40）<br>**Y：**サーバ内の RD エリア数≦ 10200 の場合：0<br>10201 ≦サーバ内の RD エリア数≦ 5672510 の場合：1<br>5672511 ≦サーバ内の RD エリア数≦ 11334820 の場合：2<br>11334821 ≦サーバ内の RD エリア数の場合：3 |
| F | pd_dbsync_point **オペランドに** commit **を指定している場合，加算します。**<br>＋ 112 ×（pd_max_users の値※× 2 ＋ 7） |
| H | **サーバ内の** raw I/O **機能を使用した** HiRDB **ファイルシステム領域に作成された** RD **エリアを格納した** HiRDB **ファイルシステム領域数が** 1001 **以上のバックエンドサーバに対して加算します。**<br>● 32 **ビットモードの場合**<br>12012 ×（↑（raw I/O 機能を使用した HiRDB ファイルシステム領域に作成された RD エリアを格納した HiRDB ファイルシステム領域数－ 1000）÷ 1000 ↑）<br>● 64 **ビットモードの場合**<br>16016 ×（↑（raw I/O 機能を使用した HiRDB ファイルシステム領域に作成された RD エリアを格納した HiRDB ファイルシステム領域数－ 1000）÷ 1000 ↑） |

注※

ディクショナリサーバの場合は pd_max_dic_process の値となります。バックエンドサーバの場合は pd_max_bes_process の値となります。ただし，pd_max_dic_process 又は pd_max_bes_process を省略する場合は，pd_max_users の値となります。

J，K，L，M，N，P，Q：固定値

この値は OS によって異なります。OS ごとの値を次に示します（単位：キロバイト）。

| OS | J **の値** | K **の値** | L **の値** | M **の値** | N **の値** | P **の値** | Q **の値** |
|---|---|---|---|---|---|---|---|
| Windows（32 ビットモード） | 75,300 | 12,000 | 15,800 | 3,600 | 5,100 | 4,400 | 4,400 |
| Windows Server 2003 (IPF) | 180,400 | 18,000 | 23,300 | 5,400 | 11,600 | 12,900 | 15,300 |
| Windows (x64) | 95,100 | 16,500 | 21,800 | 5,000 | 11,300 | 11,600 | 13,700 |

R：セキュリティ監査機能の監査証跡の絞り込み条件で指定したオブジェクトの数

S：シンクポイント出力同期制御情報取得機能使用時に必要なメモリ（単位：バイト）
pd_dbbuff_trace_level オペランドに 1 を指定し，かつ pd_dfw_awt_process オペランドの指定を省略している場合に，次の値を加算します。

32 ビットモードの場合：
320 ×シングルサーバに定義したグローバルバッファの数

64 ビットモードの場合：
640 ×シングルサーバに定義したグローバルバッファの数

T：pd_max_resident_rdarea_no オペランドに 1 以上を指定している場合に，次の値を加算します。
1648 ＋ 16 × pd_max_resident_rddarea_no の値＋ 16 × pd_max_resident_rddarea_shm_no の値

U：インメモリデータ処理で必要なメモリ所要量
計算式については，「15.2.11　インメモリデータ処理で必要なメモリ所要量」を参照してください。

V：通信トレース処理で必要なメモリ所要量

32 ビットモードの場合：
（16 ×（x － 1024）× 2）×（pd_max_server_process 値－ w）

64 ビットモードの場合：
（32 ×（x － 1024）× 2）×（pd_max_server_process 値－ w）

W：通信トレース処理で必要なメモリ所要量
ユニット内の各サーバプロセスに関して算出した下記の値

32 ビットモードの場合：
（16 ×（Z － 1024）× 2）×（最大起動プロセス数＋ 3）

64 ビットモードの場合：
（32 ×（Z － 1024）× 2）×（最大起動プロセス数＋ 3）
最大起動プロセス数については，マニュアル「HiRDB Version 8 システム定義」を参照してください。

Z：各サーバ定義として有効な pd_pth_trace_max 値です。
オペランド指定値を 2 のべき乗の値に切り上げた値になります。

表 15-8 SQL 文ごとの作業表数の求め方

| SQL 文 | 作業表数の求め方 |
|---|---|
| SELECT 文<br>INSERT（-SELECT）文 | 1. ～ 8. の指定がない場合：0<br>1. ～ 8. の指定がある場合：該当する作業表数をすべて加算した値<br>1. **複数の表を結合して検索する場合**<br>増加作業表数＝（結合表数－ 1）× 2 ＋ 1<br>2. ORDER BY **句を指定する場合**<br>増加作業表数＝ 2<br>3. GROUP BY **句を指定する場合**<br>増加作業表数＝ GROUP BY 句指定数<br>4. DISTINCT **句を指定する場合**<br>増加作業表数＝ DISTINCT 句指定数<br>5. UNION **句，**UNION ALL **句又は** EXCEPT[ALL] **句を指定する場合**<br>増加作業表数＝（UNION 又は UNION ALL 句指定数）× 2 ＋ 1<br>6. **探索条件中にインデクスを定義した列がある場合**<br>増加作業表数＝探索条件中のインデクスを定義した列数<br>7. FOR UPDATE **句又は** FOR READ ONLY **句を指定する場合**<br>増加作業表数＝ 1<br>8. **副問合せ（限定述語）を指定する場合**<br>増加作業表数＝副問合せ指定数 |
| UPDATE 文<br>DELETE 文 | 探索条件中のインデクスを定義した列数＋ 1 |
| DROP SCHEMA 文<br>DROP TABLE 文<br>DROP INDEX 文<br>CREATE INDEX 文<br>REVOKE 文でアクセス権限を取り消す場合 | 2 |

## 15.2.3 ユニットコントローラが使用する共用メモリの計算式

### （1）32 ビットモードの場合

ユニットの開始から終了までの間にユニットコントローラが使用する共用メモリは，次に示す HiRDB のプロセスの項目すべてを加算した値です。

なお，ユニットコントローラ全体の共用メモリサイズは 2 ギガバイト以内になるようにしてください。

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| スケジューラ | pd_utl_exec_mode の値が 0 の場合<br>｛↑（432 ＋ 304 × n）÷ 1024 ↑＋ 302 ＋ x ＋ z｝× 1024<br>pd_utl_exec_mode の値が 1 の場合<br>｛↑（432 ＋ 304 × n）÷ 1024 ↑＋↑（s × 2000 ＋ 136）÷ 1024 ↑＋ y ＋ z｝× 1024<br>x：ユニット内に MGR がある場合：37<br>ユニット内に FES がある場合：57 ＋ 1 ×（s ＋ 3）＋ 14<br>ユニット内に DS がある場合：102 ＋ 5 ×（t ＋ 3）＋ 14<br>ユニット内に BES がある場合：｛192 ＋ 9 ×（u ＋ 3）＋ 14｝×（BES 数＋ β ＋ γ）<br>y：ユニット内に MGR がある場合：0<br>ユニット内に FES がある場合：1 ×（s ＋ 3）＋ 14<br>ユニット内に DS がある場合：5 ×（t ＋ 3）＋ 14<br>ユニット内に BES がある場合：｛9 ×（u ＋ 3）＋ 14｝×（BES 数＋ β ＋ γ）<br>z：影響分散スタンバイレス型系切り替えの対象となるユニットの場合：<br>↑ 64 ＋｛(ユニット内 BES 数＋受け入れ可能なゲスト BES 数※）× 48｝÷ 1024 ↑<br>上記以外のユニットの場合：0<br>n：ユニット内のサーバ数＋ β ＋ γ ＋ユニット内ユティリティサーバ数＋ 1<br>ユニット内ユティリティサーバ数：24 ＋ α<br>α：ユニット内に MGR がある場合：3<br>ユニット内に FES がある場合：3<br>ユニット内に DS がある場合：7<br>ユニット内に BES がある場合：(BES 数＋ β）× 15<br>s：pd_max_users の値<br>t：pd_max_dic_process の値<br>u：pd_max_bes_process の値<br>β：影響分散スタンバイレス型系切り替えの対象となるユニットの場合：<br>受け入れ可能なゲスト BES 数※<br>上記以外のユニットの場合：0<br>γ：1：1 スタンバイレス型系切り替えの対象となるユニットの場合：代替 BES 数<br>上記以外のユニットの場合：0<br>注※　pd_ha_max_guest_act_servers の値 |
| ロックサーバ | **●ユニット内にサーバ（FES，BES，又は DS）がある場合**<br>影響分散スタンバイレス型系切り替え機能の対象となるユニットで，ゲスト BES についての計算をする場合に使用するサーバごとのオペランド（pd_lck_pool_size，pd_lck_pool_partition，pd_lck_hash_entry，pd_max_bes_process など）の値は，そのゲスト BES のオペランドの値ではなく，そのユニット内の全ゲスト BES に指定されている値の最大値を使用します。また，影響分散スタンバイレス型系切り替え機能の対象となるユニットでは，「ユニット内サーバ」は「全ホスト BES ＋全ゲスト BES」です。<br>1：1 スタンバイレス型系切り替え機能の対象となるユニットでは，「ユニット内サーバ」は「全正規 BES ＋全代替 BES」です。<br><br>$\sum_{x=1}^{y}$ ｛<br>192 ＋ 48 ＋ $c_x$ ＋ $d_x$ ＋ 48 ＋ 4096 ＋ $g_x$ ＋ 48 ＋ $i_x$<br>＋ 48 ＋ 12252 ＋ 48 ＋ $n_x$ ＋ $p_x$ ＋ $t_x$ ＋ $u_x$ ＋ 16<br>｝× pd_lck_pool_partition の値※<br><br>注※　FES の場合 pd_fes_lck_pool_partition の値<br><br>x：ユニット内サーバ通し番号<br>y：ユニット内サーバ数 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| | $c_x$：pd_lck_hash_entry を省略，又は 0 を指定している場合：<br>FES で，pd_fes_lck_pool_size が指定されていないとき<br>（↓（8 ＋ 4 × MAX（↑↓（($p_x$ ＋ 3）×（pd_max_access_tables の値＋ 4）<br>÷ pd_fes_lck_pool_partition の値↓）÷ 10 ↑，11261））÷ 16 ↓＋ 1）× 16<br>FES で，pd_fes_lck_pool_size が指定されているとき<br>（↓（8 ＋ 4 × MAX（↑（↓ pd_fes_lck_pool_size の値<br>÷ pd_fes_lck_pool_partition の値↓× 6）÷ 10 ↑，11261））<br>÷ 16 ↓＋ 1）× 16<br>BES，又は DS のとき<br>（↓（8 ＋ 4 × MAX（↑（($p_x$ ＋ 3）× 2 × 5 ＋↓ pd_lck_pool_size の値<br>÷ pd_lck_pool_partition の値↓× 6）<br>÷ 10 ↑，11261））÷ 16 ↓＋ 1）× 16<br>pd_lck_hash_entry に 2 以上の素数でない値を指定している場合：<br>（↓（8 ＋ 4 × pd_lck_hash_entry の値を超えない最大の素数）÷ 16 ↓＋ 1）<br>× 16<br>pd_lck_hash_entry に 1，又は素数を指定している場合：<br>（↓（8 ＋ 4 × pd_lck_hash_entry の値）÷ 16 ↓＋ 1）× 16<br><br>$d_x$：FES で，pd_fes_lck_pool_size が指定されていない場合：<br>（↓（($p_x$ ＋ 3）×（pd_max_access_tables の値＋ 4））<br>÷ pd_fes_lck_pool_partition の値↓）× 96<br>FES で，pd_fes_lck_pool_size が指定されている場合：<br>↓ pd_fes_lck_pool_size の値÷ pd_fes_lck_pool_partition の値↓× 6 × 96<br>BES，又は DS の場合：<br>（($p_x$ ＋ 3）× 2 × 5 ＋↓ pd_lck_pool_size の値<br>÷ pd_lck_pool_partition の値↓× 6）× 96<br>$g_x$：FES の場合：<br>（p ＋ 3）× 2 × 256<br>BES で，pd_utl_exec_mode の値 =1，かつ s ＞ 32 の場合：<br>（（p ＋ 3）× 2 ＋ s）× 256<br>BES で，pd_utl_exec_mode の値 =0，又は s ≦ 32 の場合：<br>（（p ＋ 3）× 2 ＋ 32）× 256<br>DS で，pd_utl_exec_mode の値 =1，かつ s ＞ 16 の場合：<br>（（p ＋ 3）× 2 ＋ s）× 256<br>DS で，pd_utl_exec_mode の値 =0，又は s ≦ 16 の場合：<br>（（p ＋ 3）× 2 ＋ 16）× 256<br><br>$i_x$：FES で，pd_fes_lck_pool_size が指定されていない場合：<br>（↓（($p_x$ ＋ 3）×（pd_max_access_tables の値＋ 4））<br>÷ pd_fes_lck_pool_partition の値↓）× 2 × 64<br>FES で，pd_fes_lck_pool_size が指定されている場合：<br>（↓ pd_fes_lck_pool_size の値÷ pd_fes_lck_pool_partition の値↓× 8）<br>を偶数に切り上げた値× 64<br>BES で，pd_utl_exec_mode の値 =1，かつ s ＞ 32 の場合：<br>（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 8<br>＋（($p_x$ ＋ 3）× 2 × 2 × 5 ＋ s ×（pd_max_rdarea_no の値＋ 1））<br>を偶数に切り上げた値× 64<br>BES で，pd_utl_exec_mode の値 =0，又は s ≦ 32 の場合：<br>（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 8<br>＋（($p_x$ ＋ 3）× 2 × 2 × 5 ＋ 32 ×（pd_max_rdarea_no の値＋ 1））<br>を偶数に切り上げた値× 64<br>DS で，pd_utl_exec_mode の値 =1，かつ s ＞ 16 の場合：<br>（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 8<br>＋（($p_x$ ＋ 3）× 2 × 2 × 5 ＋ s ＋ 4）を偶数に切り上げた値× 64<br>DS で，pd_utl_exec_mode の値 =0，又は s ≦ 16 の場合：<br>（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 8<br>＋（($p_x$ ＋ 3）× 2 × 2 × 5 ＋ 20）を偶数に切り上げた値× 64 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| | $n_x$：FES の場合：<br>（$p_x$ ＋ 3）× 2 × 48<br>BES で，pd_utl_exec_mode の値 =1，かつ s ＞ 32 の場合：<br>（（$p_x$ ＋ 3）× 2 × 17 ＋ s）× 80<br>BES で，pd_utl_exec_mode の値 =0，又は s ≦ 32 の場合：<br>（（$p_x$ ＋ 3）× 2 × 17 ＋ 32）× 80<br>DS で，pd_utl_exec_mode の値 =1，かつ s ＞ 16 の場合：<br>（（$p_x$ ＋ 3）× 2 × 17 ＋ s）× 80<br>DS で，pd_utl_exec_mode の値 =0，又は s ≦ 16 の場合：<br>（（$p_x$ ＋ 3）× 2 × 17 ＋ 16）× 80<br>$p_x$：FES で，HiRDB システム内の FES 数＞ 1 の場合：s ＋ 1<br>FES で，HiRDB システム内の FES 数 =1 の場合：s<br>BES で，s ＞ pd_max_bes_process の値 の場合：s<br>BES で，s ≦ pd_max_bes_process の値 の場合：pd_max_bes_process の値<br>DS で，s ＞ pd_max_dic_process の値 の場合：s<br>DS で，s ≦ pd_max_dic_process の値 の場合：pd_max_dic_process の値<br>s：pd_max_users の値<br>$t_x$：FES の場合：<br>32 ＋（$p_x$ ＋ 3）× 2 ×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 2<br>BES で，pd_utl_exec_mode の値 =1，かつ s ＞ 32 の場合：<br>32 ＋（（$p_x$ ＋ 3）× 2 ＋ s）×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 2<br>BES で，pd_utl_exec_mode の値 =0，又は s ≦ 32 の場合：<br>32 ＋（（$p_x$ ＋ 3）× 2 ＋ 32）×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 2<br>DS で，pd_utl_exec_mode の値 =1，かつ s ＞ 16 の場合：<br>32 ＋（（$p_x$ ＋ 3）× 2 ＋ s）×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 2<br>DS で，pd_utl_exec_mode の値 =0，又は s ≦ 16 の場合：<br>32 ＋（（$p_x$ ＋ 3）× 2 ＋ 16）×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 2<br>$u_x$：FES で，pd_fes_lck_pool_size が指定されていない場合：<br>32 ＋（（$p_x$ ＋ 3）×（pd_max_access_tables の値＋ 4））× 2<br>×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 4<br>FES で，pd_fes_lck_pool_size が指定されている場合：<br>32 ＋（↓ pd_fes_lck_pool_size の値÷ pd_fes_lck_pool_partition の値↓× 8）<br>を偶数に切り上げた値<br>×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 4<br>BES で，pd_utl_exec_mode の値 =1，かつ s ＞ 32 の場合：<br>32 ＋（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 8<br>＋（$p_x$ ＋ 3）× 2 × 2 × 5 ＋ s ×（pd_max_rdarea_no の値＋ 1））<br>を偶数に切り上げた値×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 4<br>BES で，pd_utl_exec_mode の値 =0，又は s ≦ 32 の場合：<br>32+（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 8<br>＋（$p_x$ ＋ 3）× 2 × 2 × 5 ＋ 32 ×（pd_max_rdarea_no の値＋ 1））<br>を偶数に切り上げた値×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 4<br>DS で，pd_utl_exec_mode の値 =1，かつ s ＞ 16 の場合：<br>32 ＋（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 8<br>＋（$p_x$ ＋ 3）× 2 × 2 × 5 ＋ s ＋ 4）を偶数に切り上げた値<br>×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 4<br>DS で，pd_utl_exec_mode の値 =0，又は s ≦ 16 の場合：<br>32 ＋（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 8<br>＋（$p_x$ ＋ 3）× 2 × 2 × 5 ＋ 20）を偶数に切り上げた値<br>×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 4<br><br>**●ユニット内にサーバ（FES，BES，又は DS）がない場合**<br>8416 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| トランザクションサーバ | 288 ＋ 32 × B ＋ 192 × s × 2<br>**ユニット内に FES がある場合に加算します**※<br>＋ 1028 ＋（420 ＋ 564 ＋ 256 ＋ 384 × 2）×（A × 2 ＋ 7）＋ 256 × 2<br>＋ 128 ×（システム内 BES 数× 4 ＋システム内 DS 数× 2<br>＋システム内 FES 数）×（A × 2 ＋ 7）<br>＋ C<br>**ユニット内に BES がある場合に加算します**※<br>＋ 1028 ＋（420 ＋ 564 ＋ 256 ＋ 384 × 2）<br>×（u × 2 ＋ 7）＋ 256 × 2<br>＋ 128 ×（システム内 BES 数× 4 ＋システム内 DS 数× 2<br>＋システム内 FES 数）×（A × 2 ＋ 7）<br>＋ D<br>**ユニット内に DS がある場合に加算します**※<br>＋ 1028 ＋（420 ＋ 564 ＋ 256 ＋ 384 × 2）<br>×（t × 2 ＋ 7）＋ 256 × 2<br>＋ 128 ×（システム内 BES 数× 4 ＋システム内 DS 数× 2<br>＋システム内 FES 数）×（A × 2 ＋ 7）<br>＋ E<br><br>s：pd_max_users の値<br>t：pd_max_dic_process の値<br>u：pd_max_bes_process の値<br>A：マルチ FES の場合：s ＋ 1<br>マルチ FES でない場合：s<br>B：影響分散スタンバイレス型系切り替えの対象となるユニットの場合：<br>ホスト BES 数＋ pd_ha_max_act_guest_servers オペランドの補正値<br>上記以外のユニットの場合：ユニット内サーバ数<br>C：≪条件≫に示すどちらかの条件に一致する場合：<br>128 ×（システム内 BES 数× 4 ＋システム内 DS 数× 2<br>＋システム内 FES 数）×（A × 2 ＋ 7）<br>≪条件≫に示すどちらの条件にも一致しない場合：0<br>D：≪条件≫に示すどちらかの条件に一致する場合：<br>128 ×（システム内 BES 数× 4 ＋システム内 DS 数× 2<br>＋システム内 FES 数）×（u × 2 ＋ 7）<br>≪条件≫に示すどちらの条件にも一致しない場合：0<br>E：≪条件≫に示すどちらかの条件に一致する場合：<br>128 ×（システム内 BES 数× 4 ＋システム内 DS 数× 2<br>＋システム内 FES 数）×（t × 2 ＋ 7）<br>≪条件≫に示すどちらの条件にも一致しない場合：0<br>≪条件≫<br>• pd_rpl_reflect_mode オペランドに uap を指定している<br>• システム内に，pdstart -k stls オペランドを指定したフロントエンドサーバが存在する<br><br>注※　上記の計算式で算出した値を各サーバの数分だけ加算します。<br><br>1：1 スタンバイレス型系切り替え適用ユニットの場合：<br>正規 BES 数＋代替 BES 数<br>上記以外の場合はユニット内サーバ数 |
| タイマサーバ | 32 ×（pd_max_users の値＋ 3）<br>×（システム内の BES 数＋ 1 ＋ユニット内ユティリティサーバ数＋ 1）<br>＋ 1440<br><br>ユニット内ユティリティサーバ数は 23 ＋ α<br>α：ユニット内に MGR がある場合：2<br>ユニット内に FES がある場合：3<br>ユニット内に DS がある場合：7<br>ユニット内に BES がある場合：BES 数× 15 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| 統計ログサーバ | 384 ＋ 128 × 16 ＋ 32 ＋ 288 × 2 ＋ 1024 ＋ 128 × 3<br>＋ pd_stj_buff_size の値× 1024 × 3 ＋ 64 ＋ 4096 ＋ 8192 |
| プロセスサーバ | 160 ＋ 512 × MAX（c，256）＋ 80 ＋ 256 ＋（pd_max_server_process の値＋ 50）×（256 ＋ 144）＋ 16 ＋ 8 × 34 ＋ 16 ＋ 16 ＋ 48 ＋ 48 ×（pd_module_trace_max の値＋ 1）<br><br>c：↑（46 ＋ d ＋ e ＋ f ＋（g ×ユニット内の BES 数※）＋ h ＋ i）÷ 16 ↑× 16<br>d：ユニット内に MGR がある場合は 59，ない場合は j<br>e：ユニット内に DS がある場合は 17，ない場合は 0<br>f：ユニット内に FES がある場合は 11，ない場合は 0<br>g：ユニット内に BES がある場合は 25，ない場合は 0<br>h：1：1 スタンバイレス型系切り替え機能を使用する場合は 9，<br>使用しない場合は 0<br>i：影響分散スタンバイレス型系切り替えの対象となるユニットの場合は 1，<br>対象でないユニットの場合は 0<br>j：pd_mlg_msg_log_unit オペランドに manager を指定しているときは 1，<br>local を指定しているときは 2<br>注※<br>1：1 スタンバイレス型系切り替えの対象となっている場合は（BES 数× 2）となります。影響分散スタンバイレス型系切り替えの対象となるユニットの場合は pd_ha_max_act_guest_servers オペランドの補正値を含みます。 |
| システムマネジャ | 640 ＋（44 ＋ 4）×（g ＋ h ＋ i）＋（100 ＋ 4）× {(p ＋ q ＋ 3）＋ u ×（15 ＋ 1)} ＋（92 ＋ 4）× c ＋ 40 ×（k ＋ m ＋ n × o ＋ u）× 14 ＋ 256 × m ＋ 128 × c ＋ 36 × d ＋ 12 × e ＋ 96 × o ＋ v ×（16 × 35 ×（k ＋ u）＋ 15 ＋ 36 × z ＋ 15）＋ w ×（48 × B ＋ 15 ＋ 4 × z ＋ 15 ＋ 4 × y ＋ 15）＋ v ×（132 ＋ 15）＋ 8 ＋ 5844 ＋ s ＋ s × o ＋ 16 ＋ 96 × o ＋ 1024 ＋ 272 × A<br><br>c：ユニット数<br>d：pdunit オペランドの -c オプション指定数<br>e：pdcltgrp オペランド指定数<br>g：システム内の FES 数<br>h：システム内の BES 数<br>i：システム内の DS 数<br>j：ユニット内の FES 数<br>k：ユニット内の BES 数<br>m：ユニット内の DS 数<br>n：ユニット内の代替 BES 数<br>o：ユニットが 1：1 スタンバイレス型系切り替えの対象となっている場合は 1，<br>対象となっていない場合は 0<br>p：i ＋ k ＋ m ＋ n<br>q：24 ＋ t ＋ j × 3 ＋ k × 15 ＋ m × 7<br>r：14 ×（k ＋ m ＋ u）＋ p ＋ q ＋ u × 15 ＋ 2 ＋ 38 ＋ 5 ＋ p × 4<br>s：212 ＋ 2052 ＋ 128 ×（r ＋ 3）＋ v ×（40 ×（k ＋ u）＋ 72 ×（k ＋ u））<br>t：ユニット内に MGR がある場合は 2，ない場合は 0<br>u：受け入れ可能なゲスト BES 数（pd_ha_max_act_guest_servers オペランドの値）<br>v：ユニットが影響分散スタンバイレス型系切り替えの対象となっている場合は 1，<br>対象となっていない場合は 0<br>w：影響分散スタンバイレス型系切り替えの対象となっているユニットがある場合は 1，<br>ない場合は 0<br>y：HA グループ内のユニット数の総和<br>z：HA グループ内のサーバ数の総和<br>A：pd_security_host_group オペランドに指定したホストに対応する IP アドレスの数<br>pd_security_host_group オペランドを指定していない場合は 0<br>B：システム共通定義に指定している pdhagroup オペランドの数 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| ネームサーバ | MAX｛65536，（X＋Y＋Z)｝＋MAX（16384，L）＋M<br><br>X：↑（256＋16＋156 ×ユニット数＋16＋16 × 126）÷ 1024 ↑× 1024<br>Y：8192<br>Z：↑（264 ×（$Z_2$＋$Z_3$＋j＋32））÷ 1024 ↑× 1024<br>$Z_2$：b＋10＋c＋11 ×自ユニット内の HiRDB サーバ数＋d＋e<br>$Z_3$：f＋7＋g＋4 ×自ユニット内の HiRDB サーバ数＋h＋i<br>L：↑（224 ×（$L_2$＋L3＋L4））÷ 1024 ↑× 1024<br>$L_2$：k＋2 ×システム内のユニット数<br>$L_3$：システム内の BES 数＋システム内の FES 数＋システム内の DS 数<br>$L_4$：15 ×システム内の HiRDB サーバ数<br>M：ユニット内の HiRDB サーバ数＋z＋m ×ユニット内システムサーバ数× 1024 ÷ 1024<br>b：ユニットに MGR がある場合：3<br>ユニットに MGR がない場合：0<br>c：1：1 スタンバイレス型系切り替えの対象となるユニットの場合：2<br>それ以外のユニットの場合：0<br>d：影響分散スタンバイレス型系切り替えの対象となるユニットの場合：<br>11 ×受け入れ可能なゲスト BES 数<br>それ以外のユニットの場合：0<br>e：1：1 スタンバイレス型系切り替えの対象となるユニットの場合：<br>6 ×自ユニット内の HiRDB サーバ数<br>それ以外のユニットの場合：0<br>f：ユニットに MGR がある場合：3<br>ユニットに MGR がない場合：0<br>g：1：1 スタンバイレス型系切り替えの対象となるユニットの場合：2<br>それ以外のユニットの場合：0<br>h：影響分散スタンバイレス型系切り替えの対象となるユニットの場合：<br>3 ×受け入れ可能なゲスト BES 数<br>それ以外のユニットの場合：0<br>i：1：1 スタンバイレス型系切り替えの対象となるユニットの場合：<br>4 ×自ユニット内の HiRDB サーバ数<br>それ以外のユニットの場合：0<br>j：ユニット内のサーバ数＋ β ＋ γ ＋ユニット内ユティリティサーバ数＋2<br>ユニット内ユティリティサーバ数：24 ＋ α<br>k：ユニットに MGR がある場合：3<br>ユニットに MGR がない場合：0<br>m：38 ＋n＋o<br>n：ユニットに MGR があり，かつユニット数が 2 以上の場合：5<br>ユニットに MGR があり，かつユニット数が 1 の場合：4<br>ユニットに MGR がなく，かつ pd_mlg_msg_log_unit の値が local の場合：4<br>ユニットに MGR がなく，かつ pd_mlg_msg_log_unit の値が manager の場合：3<br>o：4 ×ユニット内の HiRDB サーバ数<br>z：24 ＋ α<br>α：ユニット内に MGR がある場合：3<br>ユニット内に FES がある場合：3<br>ユニット内に DS がある場合：7<br>ユニット内に BES がある場合：（BES 数＋ β）× 15<br>β：影響分散スタンバイレス型系切り替えの対象となるユニットの場合：<br>受け入れ可能なゲスト BES 数※<br>上記以外のユニットの場合：0<br>γ：1：1 スタンバイレス型系切り替えの対象となるユニットの場合：代替 BES 数<br>上記以外のユニットの場合：0<br>注※　pd_ha_max_guest_act_servers の値 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| ノードマネジャ | **ユニット内に MGR がある場合**<br>↑（1152 ＋ 416 ×システムの全ユニット数＋ 80 ×システムの全サーバ数<br>＋ 1536 ＋ 928 ＋ 56 × C ＋ 240 × A ＋ 44 × A ＋ 28 × A<br>＋ 16 × B ＋ 16 ×システムの全 BES 数＋ 8 ×システムの全ユニット数<br>＋ 64 ×システムの全サーバ数＋ 32 ×システムの全サーバ数＋ 32）<br>÷ 1024 ↑× 1024<br>**ユニット内に MGR がない場合**<br>↑（928 ＋ 56 × C ＋ 240 × A ＋ 44 × A ＋ 28 × A<br>＋ 16 × B ＋ 16 ×システムの全 BES 数＋ 8 ×システムの全ユニット数<br>＋ 64 ×システムの全サーバ数＋ 32 ×システムの全サーバ数＋ 32）<br>÷ 1024 ↑× 1024<br>A：pd_utl_exec_mode=0 の場合：1024<br>pd_utl_exec_mode=1 の場合：pd_max_users の値×システムの全 BES 数× 3<br>ユニットに MGR がある場合は次に示す値を加算：pd_max_users の値× 4 ＋ 200<br>ユニットに DS がある場合は次に示す値を加算：pd_max_users の値× 3 ＋ 200<br>ユニットに BES がある場合は次に示す値を加算：pd_max_users の値× D<br>上記の計算式で算出した A の値が 1024 を超えない場合は，A を 1024 に置き換えてください。<br>B：pdcltgrp オペランドを指定しない場合：0<br>pdcltgrp オペランドを指定する場合：pdcltgrp 指定数＋ 1<br>C：ユニット内サーバ数＋ E<br>D：ユニット内 BES 数＋ E<br>E：1：1 スタンバイレス型系切り替えの対象となるユニットの場合：ユニット内の代替 BES 数<br>影響分散スタンバイレス型系切り替えの対象となるユニットの場合：受け入れ可能なゲスト BES 数<br>上記以外のユニットの場合：0 |
| I/O サーバ | ↑（28 ＋（↑（32 ＋ A）÷ 32 ↑× 32））÷ 128 ↑× 128<br>**影響分散スタンバイレス型系切り替え機能の対象となるユニットではない場合**<br>**A：pd_large_file_use=N を指定したとき**<br>3248 ＋（14 ＋ 16）× 808 ＋ 1 × 272 ＋（534 × 272）※ 1<br>＋ {(534 × 272 ＋ 16 × 272 ＋ pd_max_file_no の値× 808）× BES 数} ※ 2<br>＋ {534 × 272 ＋ 16 × 272 ＋ pd_max_file_no の値× 808} ※ 3<br>**pd_large_file_use=Y（又は省略）を指定したとき**<br>3248 ＋（14 ＋ 16）× 972 ＋ 1 × 276 ＋（534 × 276）※ 1<br>＋ {(534 × 276 ＋ 16 × 276 ＋ pd_max_file_no の値× 972）× BES 数} ※ 2<br>＋ {534 × 276 ＋ 16 × 276 ＋ pd_max_file_no の値× 972} ※ 3<br><br>注※ 1　FES がある場合に加算します。<br>注※ 2　BES がある場合に加算します。<br>注※ 3　DS がある場合に加算します。<br>なお，1：1 スタンバイレス型系切り替え構成対象ユニットの場合には，上記の式で求めた値を 2 倍にします。<br>**影響分散スタンバイレス型系切り替え機能の対象となるユニットの場合**<br>**A：pd_large_file_use=N を指定したとき**<br>↑ 48 ＋ 24 × BES 数※ 4 ÷ 16 ↑× 16<br>＋↑（3248 ＋ 16 × 808 ＋ 534 × 272 ＋ 16 × 272 ＋ pd_max_file_no の値× 808）÷ 16 ↑ × 16 × BES 数※ 4<br>＋↑（3248 ＋（14 ＋ 16）× 808 ＋ 1 × 272）÷ 16 ↑× 16<br>**pd_large_file_use=Y（又は省略）を指定したとき**<br>↑ 48 ＋ 24 × BES 数※ 4 ÷ 16 ↑× 16<br>＋↑（3248 ＋ 16 × 972 ＋ 534 × 276 ＋ 16 × 276 ＋ pd_max_file_no の値× 972）÷ 16 ↑ × 16 × BES 数※ 4<br>＋↑（3248 ＋（14 ＋ 16）× 972 ＋ 1 × 276）÷ 16 ↑× 16<br><br>注※ 4　pd_ha_max_act_guest_servers の値を含みます。 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| ログサーバ | 32 ＋ 48 ＋ 128 × 37<br>＋ {<br>384 ＋ 128 × 7 ＋ 1024 ＋ 512<br>＋↑（128 ＋ 256 ＋ 160 ＋ 8 ＋ 64）÷ pd_log_rec_leng の値※↑<br>× pd_log_rec_leng の値※<br>＋ 64 ＋ 4096 × 2 ＋（736 ＋ 512）× B<br>＋↑ {(512 ＋ 256）＋ 128 × B ＋ 464 × B} ÷（8192 － 128）↑× 8192<br>＋ 128 × pd_log_write_buff_count の値※<br>＋（pd_log_write_buff_count の値※＋ A）<br>×↑ {pd_log_max_data_size の値※＋（68 ＋ 44 ＋ 96 ＋ 160)} ÷ 4096 ↑<br>× 4096 ＋ C<br>} ×ユニット内サーバ数＋ D ＋ 128 ×ユニット内 FES 数<br>＋ 512 ×ユニット内の HiRDB のサーバ数<br><br>A：16<br>B：pdlogadfg -d sys オペランド指定数※<br>C：512<br>D：1：1 スタンバイレス型系切り替えの対象となるユニットの場合：ユニット内の代替 BES 数<br>影響分散スタンバイレス型系切り替えの対象となるユニットの場合：<br>pd_ha_max_act_guest_servers オペランドの補正値<br>注※<br>ユニット内の全サーバの指定値で，最大値を指します。1：1 スタンバイレス型系切り替えの対象となるユニットの場合はユニット内の全サーバとユニット内で動作する代替 BES の指定値で，最大値を指します。影響分散スタンバイレス型系切り替えの対象となるユニットの場合はユニット内の全サーバと HA グループに含まれる全 BES の指定値で，最大値を指します。 |
| シンクポイントダンプサーバ | {<br>↑（368 ＋ 1456 × 2）÷ 1024 ↑× 1024<br>＋↑ {(96 ＋ 80 ＋ 208 ＋ 208）＋ 192 ×（pdlogadfg -d spd の指定数※）<br>＋ 416 ×（pdlogadpf -d spd の指定数※）＋ 1023} ÷ 1024 ↑× 1024<br>} ×（全サーバ数＋ A）<br>A：1：1 スタンバイレス型系切り替えの対象となるユニットの場合：ユニット内の代替 BES 数<br>影響分散スタンバイレス型系切り替えの対象となるユニットの場合：<br>pd_ha_max_act_guest_servers オペランドの補正値<br>注※<br>ユニット内の全サーバの指定値で，最大値を指します。1：1 スタンバイレス型系切り替えの対象となるユニットの場合はユニット内の全サーバとユニット内で動作する代替 BES の指定値で，最大値を指します。影響分散スタンバイレス型系切り替えの対象となるユニットの場合はユニット内の全サーバと HA グループに含まれる全 BES の指定値で，最大値を指します。 |
| ユニット共通 | a ＋ {b ＋ 64 ＋（s ＋ 3）× c ＋ 64 ＋ 48 ＋ d ＋ e}<br>×（ユニット内の FES，BES，DS の合計数＋ i）<br>＋（g ×（ユニット内の BES，DS の合計数＋ i））＋ f<br>＋（pd_max_server_process の値＋ 100）×（48 ＋ 16）＋ 32<br>＋（pd_max_server_process の値＋ 100 ＋ 384）× 32 ＋ 32 ＋ h ＋ j<br>＋（pd_max_server_process の値＋ 127）× 32 ＋ 32<br>**影響分散スタンバイレス型系切り替え機能を使用する場合に加算します。**<br>(↑（28 ＋ ( ↑（56 ＋ 72584）÷ 32 ↑× 32)）÷ 128 ↑× 128）<br><br>a：26752 ＋ v × 4 × 34<br>b：2988<br>c：1952<br>d：32 × 32<br>e：64 ＋ 64 × {(s ＋ 3）× 2<br>＋ MAX（5，↓［s ＋ 3］÷ 10 ↓）＋ 7}<br>f：512 ×（13 ＋ユニット内の FES，BES，DS の合計数× 3）× 2<br>g：{(96 ＋ pd_lck_until_disconnect_cnt の値× 112 ＋ 4095）÷ 4096}<br>× 4096 × 2 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| | h：↑（pd_registered_port で指定したポート番号の数× 16 ＋ 32 ＋ 1023）<br>÷ 1024 ↑× 1024<br>pd_registered_port を指定していない場合は 0<br>i：1：1 スタンバイレス型系切り替えの場合は代替 BES 数<br>影響分散スタンバイレス型系切り替えの場合は pd_ha_max_guest_servers 補正値<br>j：p ×（ユニット内の FES 数）＋ q ×（ユニット内の BES 数＋ i）＋ r ×（ユニット内の DS 数）<br>k：FES の pd_fes_lck_pool_partition の値<br>m：BES の pd_lck_pool_partition の値<br>n：DS の pd_lck_pool_partition の値<br>o：(s ＋ 3）× 2 ＋ MAX｛5，↓（s ＋ 3）÷ 10 ↓｝＋ 7<br>p：k が 2 以上の場合は 32 ＋（8 ＋ 8 × k）× o<br>それ以外の場合は 0<br>q：m が 2 以上の場合は 32 ＋（8 ＋ 8 × m）× o<br>それ以外の場合は 0<br>r：n が 2 以上の場合は 32 ＋（8 ＋ 8 × n）× o<br>それ以外の場合は 0<br>s：pd_max_users の値<br>t：pd_max_dic_process の値<br>u：pd_max_bes_process の値<br>s は，DS の場合は t，BES の場合は u とします。pd_max_dic_process 又は pd_max_bes_process を省略する場合は s とします。<br>v：pd_dbbuff_modify に Y を指定し，かつユニット内に BES 又は DS がある場合：16 ＋（BES 定義又は DS 定義の pd_max_add_dbbuff_shm_no の値のユニット内最大値）＋ pd_max_resident_rdarea_shm_no 値<br>上記以外：16 ＋ pd_max_resident_rdarea_shm_no の値 |
| トランザクションログサーバ | ｛1024 ＋ 512 × A｝×（ユニット内サーバ数＋ H）<br>＋｛<br>128 × B ＋ 128<br>＋[F ＋↑（128 ＋ 256 ＋ 8 ＋ 224）÷ pd_log_rec_leng の値※↑× pd_log_rec_leng の値※<br>＋↑（pd_log_max_data_size の値※＋ 68 ＋ 44 ＋ 96 ＋ 160）<br>÷ pd_log_rec_leng の値※↑× pd_log_rec_leng の値※ ]<br>× D ＋ E ＋（48 ＋ 8）× B × 2<br>｝×（ユニット内の BES 及び DS 数＋ H）<br>＋｛<br>584 × B ＋ 128 × B ＋ 64 × B × C ＋ 128 ＋ F<br>＋↑（128 ＋ 256 ＋ 8 ＋ 224）÷ pd_log_rec_leng の値※↑× pd_log_rec_leng の値※<br>＋↑（pd_log_max_data_size の値※＋ 68 ＋ 44 ＋ 96 ＋ 160）<br>÷ pd_log_rec_leng の値※↑× pd_log_rec_leng の値※<br>＋ E ＋（48 ＋ 8）×（B × 2 ＋ 2）<br>｝×（ユニット内サーバ数＋ H） |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| | A：2<br>B：7 ＋ J × 2<br>C：システム全体の BES 数× 4 ＋システム全体の DS 数× 2<br>　＋システム全体の FES 数<br>D：pd_log_rollback_buff_count の値が 0 の場合：8<br>　それ以外の場合：pd_log_rollback_buff_count の値<br>E：512<br>F：512<br>H：1：1 スタンバイレス型系切り替えの対象となるユニットの場合：ユニット内の代替 BES 数<br>　影響分散スタンバイレス型系切り替えの対象となるユニットの場合：<br>　pd_ha_max_act_guest_servers オペランドの補正値<br>J：ユニット内サーバ中の，s，t，及び u の最大値<br>s：pd_max_users の値<br>t：pd_max_dic_process の値<br>u：pd_max_bes_process の値<br>注※<br>　ユニット内の全サーバの指定値で，最大値を指します。1：1 スタンバイレス型系切り替えの対象となるユニットの場合はユニット内の全サーバとユニット内で動作する代替 BES の指定値で，最大値を指します。影響分散スタンバイレス型系切り替えの対象となるユニットの場合はユニット内の全サーバと HA グループに含まれる全 BES の指定値で，最大値を指します。 |
| ステータスサーバ | ↑ 64 ÷ 32 ↑× 32 ×（ユニット内サーバ数＋ A）<br>A：1：1 スタンバイレス型系切り替えの対象となるユニットの場合：ユニット内の代替 BES 数<br>　影響分散スタンバイレス型系切り替えの対象となるユニットの場合：<br>　pd_ha_max_act_guest_servers オペランドの補正値 |
| 監査証跡管理サーバ | ↑ A ÷ 1024 ↑× 1024<br>A：pd_aud_file_name オペランドの指定がない場合は 640<br>　pd_aud_file_name オペランドの指定がある場合は 640 ＋（304 × 200）+B ＋ C<br>B：pd_aud_async_buff_size オペランドの値が 0 の場合は 0<br>　pd_aud_async_buff_size オペランドの値が 4096 以上の場合は次の計算値<br>　（160 × pd_aud_async_buff_count オペランドの値）<br>　+ {(↑ pd_aud_async_buff_size オペランドの値÷ 4096 ↑× 4096）<br>　× pd_aud_async_buff_count オペランドの値} ＋ 4096<br>C：pd_aud_auto_loading オペランドに Y を指定し，かつ MGR があるユニットの場合は 256 ×（システム全体のユニット数＋ 1）＋ 240<br>　上記以外の場合は 0<br><br>スタンバイレス型系切り替え機能を適用するユニットの場合，自ユニットのメモリ使用量に，系切り替え先となるユニットのセキュリティ監査のメモリ使用量を加算する必要があります。 |

## （2）64 ビットモードの場合

ユニットの開始から終了までの間にユニットコントローラが使用する共用メモリは，次に示す HiRDB のプロセスの項目すべてを加算した値です。

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| スケジューラ | pd_utl_exec_mode の値が 0 の場合<br>｛↑（432 ＋ 304 × n）÷ 1024 ↑＋ 302 ＋ x ＋ z｝× 1024<br>pd_utl_exec_mode の値が 1 の場合<br>｛↑（432 ＋ 304 × n）÷ 1024 ↑＋↑（s × 2000 ＋ 136）÷ 1024 ↑＋ y ＋ z｝× 1024<br>x：ユニット内に MGR がある場合：37<br>ユニット内に FES がある場合：57 ＋ 1 ×（s ＋ 3）＋ 14<br>ユニット内に DS がある場合：102 ＋ 5 ×（t ＋ 3）＋ 14<br>ユニット内に BES がある場合：｛192 ＋ 9 ×（u ＋ 3）＋ 14｝×（BES 数＋ β ＋ γ）<br>y：ユニット内に MGR がある場合：0<br>ユニット内に FES がある場合：1 ×（s ＋ 3）＋ 14<br>ユニット内に DS がある場合：5 ×（t ＋ 3）＋ 14<br>ユニット内に BES がある場合：｛9 ×（u ＋ 3）＋ 14｝×（BES 数＋ β ＋ γ）<br>z：影響分散スタンバイレス型系切り替えの対象となるユニットの場合：<br>↑ 64 ＋ ｛(ユニット内 BES 数＋受け入れ可能なゲスト BES 数※）× 48｝÷ 1024 ↑<br>上記以外のユニットの場合：0<br>n：ユニット内のサーバ数＋ β ＋ γ ＋ユニット内ユティリティサーバ数＋ 1<br>ユニット内ユティリティサーバ数：24 ＋ α<br>α：ユニット内に MGR がある場合：3<br>ユニット内に FES がある場合：3<br>ユニット内に DS がある場合：7<br>ユニット内に BES がある場合：(BES 数＋ β）× 15<br>s：pd_max_users の値<br>t：pd_max_dic_process の値<br>u：pd_max_bes_process の値<br>β：影響分散スタンバイレス型系切り替えの対象となるユニットの場合：<br>受け入れ可能なゲスト BES 数※<br>上記以外のユニットの場合：0<br>γ：1：1 スタンバイレス型系切り替えの対象となるユニットの場合：代替 BES 数<br>上記以外のユニットの場合：0<br>注※　pd_ha_max_guest_act_servers の値 |
| ロックサーバ | **●ユニット内にサーバ（FES，BES，又は DS）がある場合**<br>影響分散スタンバイレス型系切り替え機能の対象となるユニットで，ゲスト BES についての計算をする場合に使用するサーバごとのオペランド（pd_lck_pool_size，pd_lck_pool_partition，pd_lck_hash_entry，pd_max_bes_process など）の値は，そのゲスト BES のオペランドの値ではなく，そのユニット内の全ゲスト BES に指定されている値の最大値を使用します。また，影響分散スタンバイレス型系切り替え機能の対象となるユニットでは，「ユニット内サーバ」は「全ホスト BES ＋全ゲスト BES」です。<br>1：1 スタンバイレス型系切り替え機能の対象となるユニットでは，「ユニット内サーバ」は「全正規 BES ＋全代替 BES」です。<br><br>$\sum_{x=1}^{y}$ ｛<br>$272 + 80 + c_x + d_x + 64 + 8192 + g_x + 80 + i_x$<br>$+ 64 + 16336 + 64 + n_x + p_x + t_x + u_x + 16$<br>｝× pd_lck_pool_partition の値※<br><br>注※　FES の場合は pd_fes_lck_pool_partition の値<br><br>x：ユニット内サーバ通し番号<br>y：ユニット内サーバ数 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| | $c_x$：pd_lck_hash_entry を省略，又は 0 を指定している場合：<br>FES で，pd_fes_lck_pool_size が指定されていないとき<br>（↓（8 ＋ 8 × MAX（↑↓（($p_x$ ＋ 3）×（pd_max_access_tables の値＋ 4）<br>÷ pd_fes_lck_pool_partition の値↓）÷ 10 ↑，11261））÷ 16 ↓＋ 1）× 16<br>FES で，pd_fes_lck_pool_size が指定されているとき<br>（↓（8 ＋ 8 × MAX（↑（↓ pd_fes_lck_pool_size の値<br>÷ pd_fes_lck_pool_partition の値↓× 4）÷ 10 ↑，11261））<br>÷ 16 ↓＋ 1）× 16<br>BES，又は DS のとき<br>（↓（8 ＋ 8 × MAX（↑（($p_x$ ＋ 3）× 2 × 5<br>＋↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 4）<br>÷ 10 ↑，11261））÷ 16 ↓＋ 1）× 16<br>pd_lck_hash_entry に 2 以上の素数でない値を指定している場合：<br>（↓（8 ＋ 8 × pd_lck_hash_entry の値を超えない最大の素数）÷ 16 ↓＋ 1）<br>× 16<br>pd_lck_hash_entry に 1，又は素数を指定している場合：<br>（↓（8 ＋ 8 × pd_lck_hash_entry の値）÷ 16 ↓＋ 1）× 16<br><br>$d_x$：FES で，pd_fes_lck_pool_size が指定されていない場合：<br>（↓（($p_x$ ＋ 3）×（pd_max_access_tables の値＋ 4））<br>÷ pd_fes_lck_pool_partition の値↓）× 128<br>FES で，pd_fes_lck_pool_size が指定されている場合：<br>↓ pd_fes_lck_pool_size の値÷ pd_fes_lck_pool_partition の値↓× 4 × 128<br>BES，又は DS の場合：<br>（($p_x$ ＋ 3）× 2 × 5 ＋↓ pd_lck_pool_size の値<br>÷ pd_lck_pool_partition の値↓× 4）× 128<br>$g_x$：FES の場合：<br>（p ＋ 3）× 2 × 320<br>BES で，pd_utl_exec_mode の値 =1，かつ s ＞ 32 の場合：<br>（(p ＋ 3）× 2 ＋ s）× 320<br>BES で，pd_utl_exec_mode の値 =0，又は s ≦ 32 の場合：<br>（(p ＋ 3）× 2 ＋ 32）× 320<br>DS で，pd_utl_exec_mode の値 =1，かつ s ＞ 16 の場合：<br>（(p ＋ 3）× 2 ＋ s）× 320<br>DS で，pd_utl_exec_mode の値 =0，又は s ≦ 16 の場合：<br>（(p ＋ 3）× 2 ＋ 16）× 320 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| | $i_x$：FES で，pd_fes_lck_pool_size が指定されていない場合：<br>（↓（($p_x$ ＋ 3）×（pd_max_access_tables の値＋ 4））<br>÷ pd_fes_lck_pool_partition の値↓）× 2 × 112<br>FES で，pd_fes_lck_pool_size が指定されている場合：<br>（↓ pd_fes_lck_pool_size の値÷ pd_fes_lck_pool_partition の値↓× 5<br>＋↓（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓）<br>÷ 3 ↓）を偶数に切り上げた値× 112<br>BES で，pd_utl_exec_mode の値 =1，かつ s ＞ 32 の場合：<br>（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 5<br>＋↓（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓）÷ 3 ↓<br>＋（($p_x$ ＋ 3）× 2 × 2 × 5 ＋ s ×（pd_max_rdarea_no の値＋ 1））<br>を偶数に切り上げた値× 112<br>BES で，pd_utl_exec_mode の値 =0，又は s ≦ 32 の場合：<br>（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 5<br>＋↓（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓）÷ 3 ↓<br>＋（($p_x$ ＋ 3）× 2 × 2 × 5 ＋ 32 ×（pd_max_rdarea_no の値＋ 1））<br>を偶数に切り上げた値× 112<br>DS で，pd_utl_exec_mode の値 =1，かつ s ＞ 16 の場合：<br>（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 5<br>＋↓（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓）÷ 3 ↓<br>＋（($p_x$ ＋ 3）× 2 × 2 × 5 ＋ s ＋ 4）を偶数に切り上げた値× 112<br>DS で，pd_utl_exec_mode の値 =0，又は s ≦ 16 の場合：<br>（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 5<br>＋↓（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓）÷ 3 ↓<br>＋（($p_x$ ＋ 3）× 2 × 2 × 5 × 2 × 2 × 5 ＋ 20）<br>を偶数に切り上げた値× 112<br><br>$n_x$：FES の場合：<br>（($p_x$ ＋ 3）× 2 × 80<br>BES で，pd_utl_exec_mode の値 =1，かつ s ＞ 32 の場合：<br>（(p ＋ 3）× 2 × 17 ＋ s）× 80<br>BES で，pd_utl_exec_mode の値 =0，又は s ≦ 32 の場合：<br>（(p ＋ 3）× 2 × 17 ＋ 32）× 80<br>DS で，pd_utl_exec_mode の値 =1，かつ s ＞ 16 の場合：<br>（(p ＋ 3）× 2 × 17 ＋ s）× 80<br>DS で，pd_utl_exec_mode の値 =0，又は s ≦ 16 の場合：<br>（(p ＋ 3）× 2 × 17 ＋ 16）× 80<br>$p_x$：FES で，HiRDB システム内の FES 数＞ 1 の場合：s ＋ 1<br>FES で，HiRDB システム内の FES 数 =1 の場合：s<br>BES で，s ＞ pd_max_bes_process の値 の場合：s<br>BES で，s ≦ pd_max_bes_process の値 の場合：pd_max_bes_process の値<br>DS で，s ＞ pd_max_dic_process の値 の場合：s<br>DS で，s ≦ pd_max_dic_process の値 の場合：pd_max_dic_process の値<br><br>s：pd_max_users の値<br>$t_x$：FES の場合：<br>32+（($p_x$ ＋ 3）× 2 ×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 2<br>BES で，pd_utl_exec_mode の値 =1，かつ s ＞ 32 の場合：<br>32 ＋（(($p_x$ ＋ 3）× 2 ＋ s）×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 2<br>BES で，pd_utl_exec_mode の値 =0，又は s ≦ 32 の場合：<br>32 ＋（(($p_x$ ＋ 3）× 2 ＋ 32）×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 2<br>DS で，pd_utl_exec_mode の値 =1，かつ s ＞ 16 の場合：<br>32 ＋（(($p_x$ ＋ 3）× 2 ＋ s）×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 2<br>DS で，pd_utl_exec_mode の値 =0，又は s ≦ 16 の場合：<br>32 ＋（(($p_x$ ＋ 3）× 2 ＋ 16）×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 2 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| | $u_x$：FES で，pd_fes_lck_pool_size が指定されていない場合：<br>32 ＋（($p_x$ ＋ 3）×（pd_max_access_tables の値＋ 4））× 2<br>×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 4<br>FES で，pd_fes_lck_pool_size が指定されている場合：<br>32 ＋（↓ pd_fes_lck_pool_size の値÷ pd_fes_lck_pool_partition の値↓× 5<br>＋↓（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓）÷ 3 ↓）<br>を偶数に切り上げた値×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 4<br>BES で，pd_utl_exec_mode の値 =1，かつ s ＞ 32 の場合：<br>32 ＋（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 5<br>＋↓（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓）÷ 3 ↓<br>＋（$p_x$ ＋ 3）× 2 × 2 × 5 ＋ s ×（pd_max_rdarea_no の値＋ 1））<br>を偶数に切り上げた値×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 4<br>BES で，pd_utl_exec_mode の値 =0，又は s ≦ 32 の場合：<br>32+（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 5<br>＋↓（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓）÷ 3 ↓<br>＋（$p_x$ ＋ 3）× 2 × 2 × 5 ＋ 32 ×（pd_max_rdarea_no の値＋ 1））<br>を偶数に切り上げた値×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 4<br>DS で，pd_utl_exec_mode の値 =1，かつ s ＞ 16 の場合：<br>32 ＋（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 5<br>＋↓（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓）÷ 3 ↓<br>＋（$p_x$ ＋ 3）× 2 × 2 × 5 ＋ s ＋ 4）を偶数に切り上げた値<br>×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 4<br>DS で，pd_utl_exec_mode の値 =0，又は s ≦ 16 の場合：<br>32 ＋（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓× 5<br>＋↓（↓ pd_lck_pool_size の値÷ pd_lck_pool_partition の値↓）÷ 3 ↓<br>＋（$p_x$ ＋ 3）× 2 × 2 × 5 × 2 × 2 × 5 ＋ 20）を偶数に切り上げた値<br>×↑ pd_max_open_holdable_cursors の値÷ 16 ↑× 4<br><br>**●ユニット内にサーバ（FES，BES，又は DS）がない場合**<br>16704 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| トランザクションサーバ | 304 ＋ 32 × B ＋ 192 × s × 2<br>**ユニット内に FES がある場合に加算します**※<br>＋ 1048 ＋（416 ＋ 720 ＋ 256 ＋ 392 × 2）×（A × 2 ＋ 7）＋ 256 × 2<br>＋ 128 ×（システム内 BES 数× 4 ＋システム内 DS 数× 2<br>＋システム内 FES 数）×（A × 2 ＋ 7）<br>＋ C<br>**ユニット内に BES がある場合に加算します**※<br>＋ 1048 ＋（416 ＋ 720 ＋ 256 ＋ 392 × 2）×（u × 2 ＋ 7）<br>＋ 256 × 2<br>＋ 128 ×（システム内 BES 数× 4 ＋システム内 DS 数× 2<br>＋システム内 FES 数）×（A × 2 ＋ 7）<br>＋ D<br>**ユニット内に DS がある場合に加算します**※<br>＋ 1048 ＋（416 ＋ 720 ＋ 256 ＋ 392 × 2）×（t × 2 ＋ 7）<br>＋ 256 × 2<br>＋ 128 ×（システム内 BES 数× 4 ＋システム内 DS 数× 2<br>＋システム内 FES 数）×（A × 2 ＋ 7）<br>＋ E<br><br>s：pd_max_users の値<br>t：pd_max_dic_process の値<br>u：pd_max_bes_process の値<br>A：マルチ FES の場合は s ＋ 1，マルチ FES でない場合は s<br>B：影響分散スタンバイレス型系切り替えの対象となるユニットの場合：<br>ホスト BES 数＋ pd_ha_max_act_guest_servers オペランドの値<br>上記以外のユニットの場合：ユニット内サーバ数<br>C：≪条件≫に示すどちらかの条件に一致する場合：<br>128 ×（システム内 BES 数× 4 ＋システム内 DS 数× 2<br>＋システム内 FES 数）×（A × 2 ＋ 7）<br>≪条件≫に示すどちらの条件にも一致しない場合：0<br>D：≪条件≫に示すどちらかの条件に一致する場合：<br>128 ×（システム内 BES 数× 4 ＋システム内 DS 数× 2<br>＋システム内 FES 数）×（u × 2 ＋ 7）<br>≪条件≫に示すどちらの条件にも一致しない場合：0<br>E：≪条件≫に示すどちらかの条件に一致する場合：<br>128 ×（システム内 BES 数× 4 ＋システム内 DS 数× 2<br>＋システム内 FES 数）×（t × 2 ＋ 7）<br>≪条件≫に示すどちらの条件にも一致しない場合：0<br>≪条件≫<br>• pd_rpl_reflect_mode オペランドに uap を指定している<br>• システム内に，pdstart -k stls オペランドを指定したフロントエンドサーバが存在する<br><br>注※　上記の計算式で算出した値を各サーバの数分だけ加算します。<br>1：1 スタンバイレス型系切り替え適用ユニットの場合：<br>正規 BES 数＋代替 BES 数<br>上記以外の場合はユニット内サーバ数 |
| タイマサーバ | 32 ×（pd_max_users の値＋ 3）<br>×（システム内の BES 数＋ 1 ＋ユニット内ユティリティサーバ数＋ 1）<br>＋ 1440 ＋（48 － 32）× 2<br><br>ユニット内ユティリティサーバ数は 23 ＋ α<br>α：ユニット内に MGR がある場合：2<br>ユニット内に FES がある場合：3<br>ユニット内に DS がある場合：7<br>ユニット内に BES がある場合：BES 数× 15 |
| 統計ログサーバ | 424 ＋ 128 × 16 ＋ 32 ＋ 288 × 2 ＋ 1168 ＋ 144 × 3<br>＋ pd_stj_buff_size の値× 1024 × 3 ＋ 64 ＋ 4096 ＋ 8192 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| プロセスサーバ | 176 ＋ 528 × MAX（c，256）＋ 80 ＋ 256 ＋（pd_max_server_process の値＋ 50）×（256 ＋ 160）＋ 16 ＋ 8 × 34 ＋ 16 ＋ 16 ＋ 64 ＋ 64 ×（pd_module_trace_max の値＋ 1）<br><br>c：↑（46 ＋ d ＋ e ＋ f ＋（g ×ユニット内の BES 数※）＋ h ＋ i）÷ 16 ↑× 16<br>d：ユニット内に MGR がある場合は 59，ない場合は j<br>e：ユニット内に DS がある場合は 17，ない場合は 0<br>f：ユニット内に FES がある場合は 11，ない場合は 0<br>g：ユニット内に BES がある場合は 25，ない場合は 0<br>h：1：1 スタンバイレス型系切り替え機能を使用する場合は 9，<br>使用しない場合は 0<br>i：影響分散スタンバイレス型系切り替えの対象となるユニットの場合は 1，<br>対象でないユニットの場合は 0<br>j：pd_mlg_msg_log_unit オペランドに manager を指定しているときは 1，<br>local を指定しているときは 2<br>注※<br>1：1 スタンバイレス型系切り替えの対象となっている場合は（BES 数× 2）となります。影響分散スタンバイレス型系切り替えの対象となるユニットの場合は pd_ha_max_act_guest_servers オペランドの補正値を含みます。 |
| システムマネジャ | 672 ＋（48 ＋ 8）×（g ＋ h ＋ i）＋（108 ＋ 8）× {(p ＋ q ＋ 3) ＋ u ×（15 ＋ 1)} ＋（104 ＋ 8）× c ＋ 40 ×（k ＋ m ＋ n × o ＋ u）× 14 ＋ 256 × m ＋ 128 × c ＋ 40 × d ＋ 16 × e ＋ 96 × o ＋ v ×（16 × 35 ×（k ＋ u）＋ 15 ＋ 44 × z ＋ 15）＋ w ×（48 × B ＋ 15 ＋ 4 × z ＋ 15 ＋ 4 × y ＋ 15）＋ v ×（144 ＋ 15）＋ 8 ＋ 5864 ＋ s ＋ s × o ＋ 16 ＋ 96 × o ＋ 1024 ＋ 272 × A<br><br>c：ユニット数<br>d：pdunit オペランドの -c オプション指定数<br>e：pdcltgrp オペランド指定数<br>g：システム内の FES 数<br>h：システム内の BES 数<br>i：システム内の DS 数<br>j：ユニット内の FES 数<br>k：ユニット内の BES 数<br>m：ユニット内の DS 数<br>n：ユニット内の代替 BES 数<br>o：ユニットがスタンバイレス型系切り替えの対象となっている場合は 1，<br>対象となっていない場合は 0<br>p：i ＋ k ＋ m ＋ n<br>q：24 ＋ t ＋ j × 3 ＋ k × 15 ＋ m × 7<br>r：14 ×（k ＋ m ＋ u）＋ p ＋ q ＋ u × 15 ＋ 2 ＋ 38 ＋ 5 ＋ p × 4<br>s：236 ＋ 2052 ＋ 148 ×（r ＋ 3）＋ v ×（40 ×（k ＋ u）＋ 72 ×（k ＋ u））<br>t：ユニット内に MGR がある場合は 2，ない場合は 0<br>u：受け入れ可能なゲスト BES 数（pd_ha_max_act_guest_servers オペランドの値）<br>v：ユニットが影響分散スタンバイレス型系切り替えの対象となっている場合 1，<br>対象となっていない場合は 0<br>w：影響分散スタンバイレス型系切り替えの対象となっているユニットがある場合は 1，<br>ない場合は 0<br>y：HA グループ内のユニット数の総和<br>z：HA グループ内のサーバ数の総和<br>A：pd_security_host_group オペランドに指定したホストに対応する IP アドレスの数<br>pd_security_host_group オペランドを指定していない場合は 0<br>B：システム共通定義に指定している pdhagroup オペランドの数 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| ネームサーバ | MAX｛65536,（X＋Y＋Z)｝＋MAX（16384，L）＋M<br><br>X：↑（272 ＋ 16 ＋ 156 ×ユニット数＋ 16 ＋ 16 × 126）÷ 1024 ↑× 1024<br>Y：8192<br>Z：↑（264 ×（$Z_2$ ＋ $Z_3$ ＋ j ＋ 32））÷ 1024 ↑× 1024<br>$Z_2$：b ＋ 10 ＋ c ＋ 11 ×自ユニット内の HiRDB サーバ数＋ d ＋ e<br>$Z_3$：f ＋ 7 ＋ g ＋ 4 ×自ユニット内の HiRDB サーバ数＋ h ＋ i<br>L：↑（224 ×（$L_2$ ＋ L3 ＋ L4））÷ 1024 ↑× 1024<br>$L_2$：k ＋ 2 ×システム内のユニット数<br>$L_3$：システム内の BES 数＋システム内の FES 数＋システム内の DS 数<br>$L_4$：15 ×システム内の HiRDB サーバ数<br>M：ユニット内の HiRDB サーバ数＋ z ＋ m ×ユニット内システムサーバ数× 1024 ÷ 1024<br>b：ユニットに MGR がある場合：3<br>ユニットに MGR がない場合：0<br>c：1：1 スタンバイレス型系切り替えの対象となるユニットの場合：2<br>それ以外のユニットの場合：0<br>d：影響分散スタンバイレス型系切り替えの対象となるユニットの場合：<br>11 ×受け入れ可能なゲスト BES 数<br>それ以外のユニットの場合：0<br>e：1：1 スタンバイレス型系切り替えの対象となるユニットの場合：<br>6 ×自ユニット内の HiRDB サーバ数<br>それ以外のユニットの場合：0<br>f：ユニットに MGR がある場合：3<br>ユニットに MGR がない場合：0<br>g：1：1 スタンバイレス型系切り替えの対象となるユニットの場合：2<br>それ以外のユニットの場合：0<br>h：影響分散スタンバイレス型系切り替えの対象となるユニットの場合：<br>3 ×受け入れ可能なゲスト BES 数<br>それ以外のユニットの場合：0<br>i：1：1 スタンバイレス型系切り替えの対象となるユニットの場合：<br>4 ×自ユニット内の HiRDB サーバ数<br>それ以外のユニットの場合：0<br>j：ユニット内のサーバ数＋ β ＋ γ ＋ユニット内ユティリティサーバ数＋ 2<br>ユニット内ユティリティサーバ数：24 ＋ α<br>k：ユニットに MGR がある場合：3<br>ユニットに MGR がない場合：0<br>m：38 ＋ n ＋ o<br>n：ユニットに MGR があり，かつユニット数が 2 以上の場合：5<br>ユニットに MGR があり，かつユニット数が 1 の場合：4<br>ユニットに MGR がなく，かつ pd_mlg_msg_log_unit の値が local の場合：4<br>ユニットに MGR がなく，かつ pd_mlg_msg_log_unit の値が manager の場合：3<br>o：4 ×ユニット内の HiRDB サーバ数<br>z：24 ＋ α<br>α：ユニット内に MGR がある場合：3<br>ユニット内に FES がある場合：3<br>ユニット内に DS がある場合：7<br>ユニット内に BES がある場合：（BES 数＋ β）× 15<br>β：影響分散スタンバイレス型系切り替えの対象となるユニットの場合：<br>受け入れ可能なゲスト BES 数※<br>上記以外のユニットの場合：0<br>γ：1：1 スタンバイレス型系切り替えの対象となるユニットの場合：代替 BES 数<br>上記以外のユニットの場合：0<br>注※　pd_ha_max_guest_act_servers の値 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| ノードマネジャ | **ユニット内に MGR がある場合**<br>↑（1312 ＋ 448 ×システムの全ユニット数＋ 96 ×システムの全サーバ数<br>＋ 2048 ＋ 1104 ＋ 72 × C ＋ 240 × A ＋ 44 × A ＋ 28 × A<br>＋ 16 × B ＋ 16 ×システムの全 BES 数＋ 8 ×システムの全ユニット数<br>＋ 64 ×システムの全サーバ数＋ 32 ×システムの全サーバ数＋ 48）<br>÷ 1024 ↑× 1024<br>**ユニット内に MGR がない場合**<br>↑（1104 ＋ 72 × C ＋ 240 × A ＋ 44 × A ＋ 28 × A<br>＋ 16 × B ＋ 16 ×システムの全 BES 数＋ 8 ×システムの全ユニット数<br>＋ 64 ×システムの全サーバ数＋ 32 ×システムの全サーバ数＋ 48）<br>÷ 1024 ↑× 1024<br>A：pd_utl_exec_mode=0 の場合：1024<br>pd_utl_exec_mode=1 の場合：pd_max_users の値×システムの全 BES 数× 3<br>ユニットに MGR がある場合は次に示す値を加算：pd_max_users の値× 4 ＋ 200<br>ユニットに DS がある場合は次に示す値を加算：pd_max_users の値× 3 ＋ 200<br>ユニットに BES がある場合は次に示す値を加算：pd_max_users の値× D<br>上記の計算式で算出した A の値が 1024 を超えない場合は，A を 1024 に置き換えてください。<br>B：pdcltgrp オペランドを指定しない場合：0<br>pdcltgrp オペランドを指定する場合：pdcltgrp 指定数＋ 1<br>C：ユニット内サーバ数＋ E<br>D：ユニット内 BES 数＋ E<br>E：1：1 スタンバイレス型系切り替えの対象となるユニットの場合：ユニット内の代替 BES 数<br>影響分散スタンバイレス型系切り替えの対象となるユニットの場合：受け入れ可能なゲスト BES 数<br>上記以外のユニットの場合：0 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| I/O サーバ | ↑（56 ＋（↑（56 ＋ A）÷ 32 ↑× 32））÷ 128 ↑× 128<br>**影響分散スタンバイレス型系切り替え機能の対象となるユニットではない場合**<br>**A：pd_large_file_use=N を指定したとき**<br>3248 ＋（14 ＋ 16）× 808 ＋ 1 × 272 ＋（534 × 272）※ 1<br>＋ {(534 × 272 ＋ 16 × 272 ＋ pd_max_file_no の値× 808）× BES 数} ※ 2<br>＋ {534 × 272 ＋ 16 × 272 ＋ pd_max_file_no の値× 808} ※ 3<br>＋（48 － 32）× 3<br>**pd_large_file_use=Y（又は省略）を指定したとき**<br>3248 ＋（14 ＋ 16）× 972 ＋ 1 × 276 ＋（534 × 276）※ 1<br>＋ {(534 × 276 ＋ 16 × 276 ＋ pd_max_file_no の値× 972）× BES 数} ※ 2<br>＋ {534 × 276 ＋ 16 × 276 ＋ pd_max_file_no の値× 972} ※ 3<br>＋（48 － 32）× 3<br><br>注※ 1　FES がある場合に加算します。<br>注※ 2　BES がある場合に加算します。<br>注※ 3　DS がある場合に加算します。<br>なお，1：1 スタンバイレス型系切り替え構成対象ユニットの場合には，上記の式で求めた値を 2 倍にします。<br>**影響分散スタンバイレス型系切り替え機能の対象となるユニットの場合**<br>**A：pd_large_file_use=N を指定したとき**<br>↑ 64 ＋ 24 × BES 数※ 4 ÷ 16 ↑× 16<br>＋↑ 3296 ＋ 16 × 808 ＋ 534 × 272 ＋ 16 × 272 ＋ pd_max_file_no の値× 808）÷ 16 ↑× 16 × BES 数※ 4<br>＋↑（3296 ＋（14 ＋ 16）× 808 ＋ 1 × 272）÷ 16 ↑× 16<br>**pd_large_file_use=Y（又は省略）を指定したとき**<br>↑ 64 ＋ 24 × BES 数※ 4 ÷ 16 ↑× 16<br>＋↑（3296 ＋ 16 × 972 ＋ 534 × 276 ＋ 16 × 276 ＋ pd_max_file_no の値× 972）÷ 16 ↑× 16 × BES 数※ 4<br>＋↑（3296 ＋（14 ＋ 16）× 972 ＋ 1 × 276）÷ 16 ↑× 16<br><br>注※ 4　pd_ha_max_act_guest_servers の値を含みます。 |
| ログサーバ | 32 ＋ 48 ＋ 128 × 37<br>＋ {<br>432 ＋ 128 × 7 ＋ 1168 ＋ 512<br>＋↑（128 ＋ 256 ＋ 160 ＋ 8 ＋ 64）÷ pd_log_rec_leng の値※ ↑<br>× pd_log_rec_leng の値※<br>＋ 64 ＋ 4096 × 2 ＋（768 ＋ 512）× B<br>＋↑ {(512 ＋ 256）＋ 128 × B ＋ 464 × B} ÷（8192 － 128）↑× 8192<br>＋ 144 × pd_log_write_buff_count の値※<br>＋（pd_log_write_buff_count の値※＋ A）<br>×↑ {pd_log_max_data_size の値※＋（68 ＋ 44 ＋ 96 ＋ 160)} ÷ 4096 ↑<br>× 4096 ＋ C<br>} ×ユニット内サーバ数＋ D ＋ 128 ×ユニット内 FES 数<br>＋ 512 ×ユニット内の HiRDB のサーバ数<br><br>A：16<br>B：pdlogadfg -d sys オペランド指定数※<br>C：512<br>D：1：1 スタンバイレス型系切り替えの対象となるユニットの場合：ユニット内の代替 BES 数<br>影響分散スタンバイレス型系切り替えの対象となるユニットの場合：<br>pd_ha_max_act_guest_servers オペランドの補正値<br>注※<br>ユニット内の全サーバの指定値で，最大値を指します。1：1 スタンバイレス型系切り替えの対象となるユニットの場合はユニット内の全サーバとユニット内で動作する代替 BES の指定値で，最大値を指します。影響分散スタンバイレス型系切り替えの対象となるユニットの場合はユニット内の全サーバと HA グループに含まれる全 BES の指定値で，最大値を指します。 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| シンクポイントダンプサーバ | {<br>↑（384 ＋ 1536 × 2）÷ 1024 ↑× 1024<br>＋↑ {(128 ＋ 80 ＋ 240 ＋ 240）＋ 192 ×（pdlogadfg -d spd の指定数※）<br>＋ 416 ×（pdlogadpf -d spd の指定数※）＋ 1023} ÷ 1024 ↑× 1024<br>} ×（全サーバ数＋ A）<br>A：1：1 スタンバイレス型系切り替えの対象となるユニットの場合：ユニット内の代替 BES 数<br>影響分散スタンバイレス型系切り替えの対象となるユニットの場合：<br>pd_ha_max_act_guest_servers オペランドの補正値<br>注※<br>ユニット内の全サーバの指定値で，最大値を指します。1：1 スタンバイレス型系切り替えの対象となるユニットの場合はユニット内の全サーバとユニット内で動作する代替 BES の指定値で，最大値を指します。影響分散スタンバイレス型系切り替えの対象となるユニットの場合はユニット内の全サーバと HA グループに含まれる全 BES の指定値で，最大値を指します。 |
| ユニット共通 | a ＋ {b ＋ 80 ＋（s ＋ 3）× c ＋ 64 ＋ 48 ＋ d ＋ e}<br>×（ユニット内の FES，BES，DS の合計数＋ i）<br>＋（g ×（ユニット内の BES，DS の合計数＋ i））＋ f<br>＋（pd_max_server_process の値＋ 100）×（64 ＋ 16）＋ 32<br>＋（pd_max_server_process の値＋ 100 ＋ 384）× 32 ＋ 32 ＋ h ＋ j<br>＋（pd_max_server_process の値＋ 127）× 48 ＋ 32<br>**影響分散スタンバイレス型系切り替え機能を使用する場合に加算します。**<br>（↑（56 ＋（↑（56 ＋ 88560）÷ 32 ↑× 32））÷ 128 ↑× 128）<br><br>a：36160 ＋ v × 4 × 34<br>b：3480<br>c：2760<br><br>d：48 × 32<br>e：80 ＋ 96 × {(s ＋ 3）× 2<br>＋ MAX（5，↓［s ＋ 3］÷ 10 ↓）＋ 7}<br>f：512 ×（13 ＋（ユニット内の FES，BES，DS の合計数＋ i）× 3）× 2<br>g：{(128 ＋ pd_lck_until_disconnect_cnt の値× 112 ＋ 4095）÷ 4096}<br>× 4096 × 2<br>h：↑（pd_registered_port で指定したポート番号の数× 16 ＋ 32 ＋ 1023）<br>÷ 1024 ↑× 1024<br>pd_registered_port を指定していない場合は 0<br>i：1：1 スタンバイレス型系切り替えの場合は代替 BES 数<br>影響分散スタンバイレス型系切り替えの場合は pd_ha_max_guest_servers 補正値<br>j：p ×（ユニット内の FES 数）＋ q ×（ユニット内の BES 数＋ i）＋ r ×（ユニット内の DS 数）<br><br>k：FES の pd_fes_lck_pool_partition の値<br>m：BES の pd_lck_pool_partition の値<br>n：DS の pd_lck_pool_partition の値<br>o：(s ＋ 3）× 2 ＋ MAX {5，↓（s ＋ 3）÷ 10 ↓} ＋ 7<br>p：k が 2 以上の場合は 32 ＋（8 ＋ 16 × k）× o<br>それ以外の場合は 0<br>q：m が 2 以上の場合は 32 ＋（8 ＋ 16 × m）× o<br>それ以外の場合は 0<br>r：n が 2 以上の場合は 32 ＋（8 ＋ 16 × n）× o<br>それ以外の場合は 0<br>s：pd_max_users の値<br>t：pd_max_dic_process の値<br>u：pd_max_bes_process の値<br>s は，DS の場合は t，BES の場合は u，pd_max_dic_process 又は pd_max_bes_process を省略する場合は s とします。<br>v：pd_dbbuff_modify に Y を指定し，かつユニット内に BES 又は DS がある場合：16 ＋（BES 定義又は DS 定義の pd_max_add_dbbuff_shm_no の値のユニット内最大値）＋ pd_max_resident_rdarea_shm_no 値<br>上記以外：16 ＋ pd_max_resident_rdarea_shm_no の値 |

| プロセスの種類 | 共用メモリの計算式（単位：バイト） |
|---|---|
| トランザクションログサーバ | {1168 ＋ 688 × A} ×（ユニット内サーバ数＋ H）<br>＋ {<br>128 × B ＋ 144<br>＋ [G ＋↑（128 ＋ 256 ＋ 8 ＋ 224）÷ pd_log_rec_leng の値※↑× pd_log_rec_leng の値※<br>＋↑（pd_log_max_data_size の値※＋ 68 ＋ 44 ＋ 96 ＋ 160）<br>÷ pd_log_rec_leng の値※↑× pd_log_rec_leng の値※ ]<br>× D ＋ E ＋（48 ＋ 8）× B × 2<br>} ×（ユニット内の BES 及び DS 数＋ H）<br>＋ {<br>600 × B ＋ 128 × B ＋ 64 × B × C ＋ 144 ＋ G<br>＋↑（128 ＋ 256 ＋ 8 ＋ 224）÷ pd_log_rec_leng の値※↑× pd_log_rec_leng の値※<br>＋↑（pd_log_max_data_size の値※＋ 68 ＋ 44 ＋ 96 ＋ 160）<br>÷ pd_log_rec_leng の値※↑× pd_log_rec_leng の値※<br>＋ E ＋（48 ＋ 8）×（B × 2 ＋ 2）<br>} ×（ユニット内サーバ数＋ H）<br><br>A：2<br>B：7 ＋ J × 2<br>C：システム全体の BES 数× 4 ＋システム全体の DS 数× 2<br>＋システム全体の FES 数<br>D：pd_log_rollback_buff_count の値が 0 の場合：8<br>それ以外の場合：pd_log_rollback_buff_count の値<br>E：512<br>G：512<br>H：1：1 スタンバイレス型系切り替えの対象となるユニットの場合：ユニット内の代替 BES 数<br>影響分散スタンバイレス型系切り替えの対象となるユニットの場合：<br>pd_ha_max_act_guest_servers オペランドの補正値<br><br>J：ユニット内サーバ中の，s，t，及び u の最大値<br>s：pd_max_users の値<br>t：pd_max_dic_process の値<br>u：pd_max_bes_process の値<br>注※<br>ユニット内の全サーバの指定値で，最大値を指します。1：1 スタンバイレス型系切り替えの対象となるユニットの場合はユニット内の全サーバとユニット内で動作する代替 BES の指定値で，最大値を指します。影響分散スタンバイレス型系切り替えの対象となるユニットの場合はユニット内の全サーバと HA グループに含まれる全 BES の指定値で，最大値を指します。 |
| ステータスサーバ | ↑ 64 ÷ 32 ↑× 32 ×（ユニット内サーバ数＋ A）<br>A：1：1 スタンバイレス型系切り替えの対象となるユニットの場合：ユニット内の代替 BES 数<br>影響分散スタンバイレス型系切り替えの対象となるユニットの場合：<br>pd_ha_max_act_guest_servers オペランドの補正値 |
| 監査証跡管理サーバ | ↑ A ÷ 1024 ↑× 1024<br>A：pd_aud_file_name オペランドの指定がない場合は 704<br>pd_aud_file_name オペランドの指定がある場合は 704 ＋（320 × 200）+B ＋ C<br>B：pd_aud_async_buff_size オペランドの値が 0 の場合は 0<br>pd_aud_async_buff_size オペランドの値が 4096 以上の場合は次の計算値<br>（176 × pd_aud_async_buff_count オペランドの値）<br>+ {(↑ pd_aud_async_buff_size オペランドの値÷ 4096 ↑× 4096)<br>× pd_aud_async_buff_count オペランドの値} ＋ 4096<br>C：pd_aud_auto_loading オペランドに Y を指定し，かつ MGR があるユニットの場合は 256 ×（システム全体のユニット数＋ 1）＋ 256<br>上記以外の場合は 0<br><br>スタンバイレス型系切り替え機能を適用するユニットの場合，自ユニットのメモリ使用量に，系切り替え先となるユニットのセキュリティ監査のメモリ使用量を加算する必要があります。 |

## 15.2.4 各サーバが使用する共用メモリの計算式

### (1) フロントエンドサーバが使用する共用メモリの計算式

フロントエンドサーバが使用する共用メモリの計算式を次に示します。計算式で使用している変数については，「(4) 計算式で使用する変数」を参照してください。

**● 32 ビットモードの場合**

```
40 + b × 1.3 + c + d + k + 1.6 + x + y + 4
+ {[ (a + 12) ÷ 13] × 1.1 + [ (a + 62) ÷ 63] + 3.7} × (e + 3)
+ {
    ↑↑ b ÷ 64 ↑× (8 ÷ 16) ↑× 4 × 4
    + 12 × {(b ÷ 3) + 1 − mod (b ÷ 3, 2)}
    + 4 × a × {(e + 3) × 2 + 1 + MAX (e ÷ 10, 5)}
    + 36 + {28 × (f + 1) × g} + 20000 + 112 × B
    +↑ {(c ÷ 8) + 7} ÷ 64 ↑× 8 +↑ {(k ÷ 8) + 7} ÷ 64 ↑× 8
    + MAX {a × (e + 3), c ÷ 8} × 104 + MAX {a × (e + 3), k ÷ 8} × 24
    +↑ {(x ÷ 4) + 7} ÷ 64 ↑× 8
    +↑ {[ (y − (s × 592 + t × 916 + u × 172)) ÷ 2] + 7} ÷ 64 ↑× 8
    + MAX {13 × (e + 3), x ÷ 4} × 88
    + 60 × MAX {21 × (e + 3), (y − (s × 592 + t × 916 + u × 172)) ÷ 2}
    + 44 + 256 + 1024
  } ÷ 1024 + A + 7
   I
+ Σ (Ji)
  i=1
● pd_def_buf_control_area_assign オペランドに INITIAL を指定するか，又はこのオペランドを省略した場合に加算します。
+ {[(a + 12) ÷ 13] × 1.1 + [(a + 62) ÷ 63] + 3.7} × (e + 7)
● HiRDB External Data Access 機能を使用する場合に加算します。
+↑ {↑ (40 + (32 + 144 × C + D)) ÷ 512 ↑× 512} ÷ 1024 ↑ + ↑ (B × 3616) ÷ 1024 ↑
                                                    (単位：キロバイト)
```

**● 64 ビットモードの場合**

```
40 + b × 1.3 + c + d + k + 1.6 + x + y + 5
+ {[ (a + 12) ÷ 13] × 1.2 + [ (a + 62) ÷ 63] × 1.5 + 4.1} × (e + 3)
+ {
    ↑↑ b ÷ 64 ↑× (8 ÷ 16) ↑× 4 × 4
    + 12 × {(b ÷ 3) + 1 − mod (b ÷ 3, 2)}
    + 4 × a × {(e + 3) × 2 + 1 + MAX (e ÷ 10, 5)}
    + 56 + {40 × (f + 1) × g} + 20000 + 112 × B
    +↑ {(c ÷ 8) + 7} ÷ 64 ↑× 8 +↑ {(k ÷ 8) + 7} ÷ 64 ↑× 8
    + MAX {a × (e + 3), c ÷ 8} × 104 + MAX {a × (e + 3), k ÷ 8} × 40
    +↑ {(x ÷ 4) + 7} ÷ 64 ↑× 8
    +↑ {[ (y − (s × 600 + t × 936 + u × 184)) ÷ 2] + 7} ÷ 64 ↑× 8
    + MAX {13 × (e + 3), x ÷ 4} × 104
    + 72 × MAX {21 × (e + 3), (y − (s × 600 + t × 936 + u × 184)) ÷ 2}
    + 72 + 256 + 1536
  } ÷ 1024 + A + 7
   I
+ Σ (Ji)
  i=1
● pd_def_buf_control_area_assign オペランドに INITIAL を指定するか，又はこのオペランドを省略した場合に加算します。
+ {[(a + 12) ÷ 13] × 1.2 + [(a + 62) ÷ 63] × 1.5 + 4.1} × (e + 7)
                                                    (単位：キロバイト)
```

### (2) ディクショナリサーバが使用する共用メモリの計算式

ディクショナリサーバが使用する共用メモリの計算式を次に示します。

**32 ビットモードの場合**（単位：キロバイト）

計算式 1 ＋↑｛(↑（40 ＋（計算式 2 ～計算式 5 を加算した値））÷ 512 ↑× 512)｝÷ 1024 ↑

**64 ビットモードの場合**（単位：キロバイト）

計算式 1 ＋↑｛(↑（72 ＋（計算式 2 ～計算式 5 を加算した値））÷ 512 ↑× 512)｝÷ 1024 ↑

計算式で使用している変数については，「(4) 計算式で使用する変数」を参照してください。

**注意事項**

- pd_dbsync_point 又は pd_system_dbsync_point オペランドのどちらかの指定が commit の場合に計算式 3 を加算します。pd_system_dbsync_point オペランドの省略値は commit です。
- pd_dfw_awt_process オペランドを指定する場合に計算式 4 を加算します。
- pd_max_commit_write_reclaim_no オペランドを省略した場合（ただし，pd_sysdef_default_option オペランドに ” v6compatible ” 又は ” v7compatible ” を指定した場合は除きます），又は pd_max_commit_write_reclaim_no オペランドに 0 以外を指定した場合に計算式 5 を加算します。ただし，計算式 3 を加算している場合には加算する必要はありません。
- pd_dic_shmpool_size オペランドを省略すると，次の値が設定されます。
  32 ビットモードの場合：
  ↑｛(↑（40 ＋（計算式 2 ～計算式 5 の合計値））÷ 512 ↑× 512)｝÷ 1024 ↑
  64 ビットモードの場合：
  ↑｛(↑（72 ＋（計算式 2 ～計算式 5 の合計値））÷ 512 ↑× 512)｝÷ 1024 ↑

| 条件 | 共用メモリの計算式 |
|---|---|
| 計算式 1<br>（単位：キロバイト） | ● **32 ビットモードの場合**<br>b × 1.3<br>＋｛<br>↑↑ b ÷ 64 ↑×（8 ÷ 16）↑× 4 × 4<br>＋ 12 ×｛(b ÷ 3）＋ 1 － mod（b ÷ 3，2)｝<br>＋ 8 × a ×｛(e ＋ 3）× 2 ＋ 1 ＋ MAX（e ÷ 10，5)｝<br>＋ 512<br>｝÷ 1024<br>＋ 3.5 ＋↑（224 × v × w）÷ 1024 ↑＋ 0.5<br>＋↑｛<br>↑（28 ＋（↑（32 ＋（(↑ g ÷ 127 ↑＋ 1）× 2048 ＋ 128)）÷ 32 ↑× 32))<br>÷ 128 ↑× 128<br>｝÷ 1024 ↑<br>K<br>＋ Σ（Li）<br>i=1 |

| 条件 | 共用メモリの計算式 |
|---|---|
| | ● 64 ビットモードの場合<br>b × 1.3<br>＋ {<br>↑↑ b ÷ 64 ↑×（8 ÷ 16）↑× 4 × 4<br>＋ 12 ×｛(b ÷ 3) ＋ 1 － mod（b ÷ 3，2)｝<br>＋ 8 × a ×｛(e ＋ 3) × 2 ＋ 1 ＋ MAX（e ÷ 10，5)｝<br>＋ 1024<br>｝÷ 1024<br>＋ 3.5 ＋↑（224 × v × w）÷ 1024 ↑＋ 0.5<br>＋↑ {<br>↑（56 ＋（↑（56 ＋（(↑ g ÷ 127 ↑＋ 1) × 2048 ＋ 128)）÷ 32 ↑× 32））<br>÷ 128 ↑× 128<br>｝÷ 1024 ↑<br>K<br>＋ Σ（Li）<br>i=1 |
| 計算式 2<br>（単位：バイト） | ● 32 ビットモードの場合<br>500 × 1024<br>＋（↑ 372 × g ÷ 16 ↑× 16）＋ 328 × h ＋ 112 × 240 ※ 3<br>＋ 5072 ×（e ＋ 15）＋ 96 × z<br>＋ 32 × m ＋ 172 ×｛a ×（e ＋ 3）＋（e ＋ 3）× 2 ＋ 22｝＋ 16<br>＋ 48 × p ＋ 36 ×｛(e ＋ 3) × 2 ＋ 1 ＋ MAX（5，[e ＋ 3] ÷ 10)｝<br>＋ 68 × G ＋ 64 ※ 1 ＋ 368 ※ 2<br>＋（(↓（↑（g ÷ 8）↑＋ 3）÷ 4 ↓）× 4）× m<br>● 64 ビットモードの場合<br>500 × 1024<br>＋（↑ 472 × g ÷ 16 ↑× 16）＋ 344 × h<br>＋（↑ 136 × 240 ÷ 16 ↑× 16）※ 4<br>＋ 9424 ×（e ＋ 15）＋ 144 × z<br>＋ 48 × m ＋ 336 ×｛a ×（e ＋ 3）＋（e ＋ 3）× 2 ＋ 22｝＋ 16<br>＋ 64 × p ＋ 72 ×｛(e ＋ 3) × 2 ＋ 1 ＋ MAX（5，[e ＋ 3] ÷ 10)｝<br>＋ 68 × G ＋ 64 ※ 1 ＋ 448 ※ 2<br>＋（(↓（↑（g ÷ 8）↑＋ 7）÷ 8 ↓）× 8）× m |
| 計算式 3<br>（単位：バイト） | ● 32 ビットモードの場合<br>(32 ＋ 16 × z) ×（G ＋ 1）＋ 16<br>● 64 ビットモードの場合<br>(48 ＋ 32 × z) ×（G ＋ 1）＋ 16 |
| 計算式 4<br>（単位：バイト） | ● 32 ビットモードの場合<br>72 ＋ 52 × H ＋ 68 × z<br>pd_dbbuff_trace_level オペランドに 1 を指定する場合に加算します。<br>＋ 320 × z<br>● 64 ビットモードの場合<br>96 ＋ 56 × H ＋ 72 × z<br>pd_dbbuff_trace_level オペランドに 1 を指定する場合に加算します。<br>＋ 640 × z |
| 計算式 5<br>（単位：バイト） | ● 32 ビットモードの場合<br>(32 ＋ 16 × z) × P ＋ 16<br>● 64 ビットモードの場合<br>(48 ＋ 32 × z) × P ＋ 16 |

注※ 1

pd_max_ard_process オペランドに 1 以上を指定した場合に加算します。

注※ 2

再編成時期予測機能を使用する場合に加算します。

注※ 3

pd_sysdef_default_option オペランドに ” v6compatible ” 又は ” v7compatible ” を指定した場合，112 × 240 の替わりに 112 × MAX（r，i × 1.2）で計算してください。

注※ 4

pd_sysdef_default_option オペランドに ” v6compatible ” 又は ” v7compatible ” を指定した場合，（↑ 136 × 240 ÷ 16 ↑ × 16）の替わりに（↑ 136 × MAX（r，（i × 1.2））÷ 16 ↑ × 16）で計算してください。

## (3) バックエンドサーバが使用する共用メモリの計算式

バックエンドサーバが使用する共用メモリの計算式を次に示します。

**32 ビットモードの場合**（単位：キロバイト）

計算式 1 ＋↑ {(↑（40 ＋（計算式 2 ～計算式 4，計算式 6，計算式 8 を加算した値））÷ 512 ↑ × 512)} ÷ 1024 ↑＋↑ {(↑（40 ＋計算式 5）÷ 512 ↑ × 512)} ÷ 1024 ↑

**64 ビットモードの場合**（単位：キロバイト）

計算式 1 ＋↑ {(↑（72 ＋（計算式 2 ～計算式 4，計算式 6，計算式 8 を加算した値））÷ 512 ↑ × 512)} ÷ 1024 ↑

計算式で使用している変数については，「(4) 計算式で使用する変数」を参照してください。

**計算式 1 ～ 8 についての注意事項**

- 次に示すどれかの条件を満たす場合に計算式 3 を加算します。
  - ・pd_rdarea_open_attribute_use オペランドに Y を指定する場合
  - ・高速系切り替え機能を使用する場合
- 次に示すどれかの条件を満たす場合に計算式 4 を加算します。
  - ・pd_dbsync_point オペランドに commit を指定する場合
  - ・pd_shared_rdarea_use オペランドに Y を指定する場合
- HiRDB External Data Access をインストールする場合に計算式 5 を加算します。
- pd_dfw_awt_process オペランドを指定する場合に計算式 6 を加算します。
- pd_max_commit_write_reclaim_no オペランドを省略した場合（ただし，pd_sysdef_default_option オペランドに ” v6compatible ” 又は ” v7compatible ” を指定した場合は除きます），又は pd_max_commit_write_reclaim_no オペランドに 0 以外を指定した場合に計算式 7 を加算します。ただし，計算式 4 を加算している場合には加算する必要はありません。
- pd_bes_shmpool_size オペランドを省略すると，次の値が設定されます。
  32 ビットモードの場合：
  ↑ {(↑（40 ＋（計算式 2 ～計算式 4，計算式 6，計算式 8 の合計値））÷ 512 ↑ × 512)} ÷ 1024 ↑＋↑ {(↑（40 ＋計算式 5）÷ 512 ↑ × 512)} ÷ 1024 ↑
  64 ビットモードの場合：
  ↑ {(↑（72 ＋（計算式 2 ～計算式 4，計算式 6，計算式 8 の合計値））÷ 512 ↑ × 512)} ÷ 1024 ↑
- pd_max_resident_rdarea_no オペランドを指定する場合に計算式 8 を加算します。

| 条件 | 共用メモリの計算式 |
|---|---|
| 計算式 1<br>（単位：キロバイト） | **● 32 ビットモードの場合**<br>b × 1.3<br>＋ {<br>↑↑ b ÷ 64 ↑×（8 ÷ 16）↑× 4 × 4<br>＋ 12 ×｛(b ÷ 3) ＋ 1 − mod（b ÷ 3, 2)｝<br>＋ 8 × a ×｛(e ＋ 3) × 2 ＋ 1 ＋ MAX（e ÷ 10, 5)｝＋ 512 ＋ 512 ※1<br>｝÷ 1024<br>＋↑｛72 ＋ 8 × v ×（8 ＋ 3 × w)｝÷ 1024 ↑<br>＋↑｛(↑ g ÷ 127 ↑＋ 1) × 2048 ＋ 128｝÷ 1024 ↑<br>＋↑ {<br>↑（28 ＋（↑（32 ＋（(↑ g ÷ 127 ↑＋ 1) × 2048 ＋ 128)）÷ 32 ↑× 32))<br>÷ 128 ↑× 128｝<br>｝÷ 1024 ↑<br>M<br>＋ Σ（Ni）<br>i=1<br>**● 64 ビットモードの場合**<br>b × 1.3<br>＋ {<br>↑↑ b ÷ 64 ↑×（8 ÷ 16）↑× 4 × 4<br>＋ 12 ×｛(b ÷ 3) ＋ 1 − mod（b ÷ 3, 2)｝<br>＋ 8 × a ×｛(e ＋ 3) × 2 ＋ 1 ＋ MAX（e ÷ 10, 5)｝＋ 1024 ＋ 512 ※1<br>｝÷ 1024<br>＋↑｛72 ＋ 24 × v ×（2 ＋ w)｝÷ 1024 ↑<br>＋↑ {<br>↑（56 ＋（↑（56 ＋（(↑ g ÷ 127 ↑＋ 1) × 2048 ＋ 128)）÷ 32 ↑× 32))<br>÷ 128 ↑× 128<br>｝÷ 1024 ↑<br>M<br>＋ Σ（Ni）<br>i=1 |
| 計算式 2<br>（単位：バイト） | **● 32 ビットモードの場合**<br>500 × 1024<br>＋（308 ＋ 48 ※1）× g ＋ 328 × h ＋ 112 × r ※3<br>＋ 5072 ×（e ＋ 15）＋ 96 × z<br>＋ 32 × m ＋ 172 ×｛a ×（e ＋ 3）＋（e ＋ 3）× 2 ＋ 22｝＋ 16<br>＋ 48 × p ＋ 48 ×｛(e ＋ 3) × 2 ＋ 1 ＋ MAX（5, [e ＋ 3] ÷ 10)｝<br>＋ 68 × G ＋ 144 × F ＋ 80 ＋ 32 × g ＋ 64 ※2<br>＋（(↓（↑（g ÷ 8）↑＋ 3）÷ 4 ↓）× 4）× m<br>**● 64 ビットモードの場合**<br>500 × 1024<br>＋（400 ＋ 56 ※1）× g ＋ 344 × h ＋ 136 × r ※4<br>＋ 9424 ×（e ＋ 15）＋ 144 × z<br>＋ 48 × m ＋ 336 ×｛a ×（e ＋ 3）＋（e ＋ 3）× 2 ＋ 22｝＋ 16<br>＋ 64 × p ＋ 96 ×｛(e ＋ 3) × 2 ＋ 1 ＋ MAX（5, [e ＋ 3] ÷ 10)｝<br>＋ 68 × G ＋ 160 × F ＋ 96 ＋ 48 × g ＋ 64 ※2<br>＋（(↓（↑（g ÷ 8）↑＋ 7）÷ 8 ↓）× 8）× m |
| 計算式 3<br>（単位：バイト） | **● 32 ビットモードの場合**<br>{{[（↑↑ g ÷ 8 ↑÷ 4 ↑）× 4] ＋ 8｝×（a × [e ＋ 3] ＋ e ＋ 15)｝<br>**● 64 ビットモードの場合**<br>{{[（↑↑ g ÷ 8 ↑÷ 8 ↑）× 8] ＋ 8｝×（a × [e ＋ 3] ＋ e ＋ 15)｝ |

| 条件 | 共用メモリの計算式 |
|---|---|
| 計算式 4<br>（単位：バイト） | ● 32 ビットモードの場合<br>（32 ＋ 16 × z）×（e × 2 ＋ 7 ＋ 1）＋ 16<br>● 64 ビットモードの場合<br>（48 ＋ 32 × z）×（e × 2 ＋ 7 ＋ 1）＋ 16 |
| 計算式 5<br>（単位：バイト） | ● 32 ビットモードの場合<br>32 ＋ 144 × C ＋（6160 ＋ 112 ＋ 48 × e）× B |
| 計算式 6<br>（単位：バイト） | ● 32 ビットモードの場合<br>72 ＋ 52 × H ＋ 68 × z<br>pd_dbbuff_trace_level オペランドに 1 を指定する場合に加算します。<br>＋ 320 × z<br>● 64 ビットモードの場合<br>96 ＋ 56 × H ＋ 72 × z<br>pd_dbbuff_trace_level オペランドに 1 を指定する場合に加算します。<br>＋ 640 × z |
| 計算式 7<br>（単位：バイト） | ● 32 ビットモードの場合<br>（32 ＋ 16 × z）× P ＋ 16<br>● 64 ビットモードの場合<br>（48 ＋ 32 × z）× P ＋ 16 |
| 計算式 8<br>（単位：バイト） | 16 ＋ 112 ＋（48 ＋ 48 × Q）＋（48 ＋ 32 × R） |

注※ 1

pd_max_list_users 及び pd_max_list_count オペランドの両方とも 0 でない場合に加算します。

注※ 2

pd_max_ard_process オペランドに 1 以上を指定した場合に加算します。

注※ 3

pd_sysdef_default_option オペランドに ” v6compatible ” 又は ” v7compatible ” を指定した場合，112 × r の替わりに 112 × MAX（(i × 1.2)，r）で計算してください。

注※ 4

pd_sysdef_default_option オペランドに ” v6compatible ” 又は ” v7compatible ” を指定した場合，136 × r の替わりに 136 × MAX（(i × 1.2)，r）で計算してください。

## (4) 計算式で使用する変数

a：pd_max_access_tables オペランドの値

b：pd_sql_object_cache_size オペランドの値

c：pd_table_def_cache_size オペランドの値

d：pd_auth_cache_size オペランドの値

e：pd_max_users オペランドの値 ※ 1

f：バックエンドサーバの総数

g：pd_max_rdarea_no オペランドの値

h：pd_max_file_no オペランドの値

ｉ：サーバ内のインデクスの総数
ただし，分割キーインデクスの場合はサーバ内分割数になります。

ｋ：pd_view_def_cache_size オペランドの値

ｍ：インデクス用のグローバルバッファプール数
pd_dbbuff_modify オペランドに Y を指定している場合，該当するサーバに関連する pdbuffer 文の数に，該当するサーバ定義の pd_max_add_dbbuff_no オペランドの値を加算して計算します。

ｐ：pd_lck_until_disconnect_cnt オペランドの値

ｑ：MIN（e ＋ 3，p）

ｒ：pd_assurance_index_no オペランドの値

ｓ：インストールしたプラグインの数

ｔ：DML で使用するプラグイン関数の総数 ※2

ｕ：DML で使用するプラグイン関数のパラメタ総数 ※2

ｖ：pd_max_list_users オペランドの値

ｗ：pd_max_list_count オペランドの値

ｘ：pd_type_def_cache_size オペランドの値

ｙ：pd_routine_def_cache_size オペランドの値

ｚ：グローバルバッファ総数（pdbuffer オペランドの指定数）
pd_dbbuff_modify オペランドに Y を指定している場合，該当するサーバに関連する pdbuffer 文の数に，該当するサーバ定義の pd_max_add_dbbuff_no オペランドの値を加算して計算します。

Ａ：pd_registry_cache_size オペランドの値

Ｂ：pd_max_foreign_server オペランドの値

Ｃ：HiRDB External Data Access Adapter のサポート対象の DBMS の総数
- HiRDB 又は XDM/RD E2 接続用の HiRDB External Data Access Adapter の場合：3
- ORACLE 接続用の HiRDB External Data Access Adapter の場合：1

すべての HiRDB External Data Access Adapter をインストールした場合は，サポート対象の DBMS の総数が 5 になります。

Ｄ：HiRDB External Data Access Adapter が SQL 変換で使用する領域の総和
- HiRDB 又は XDM/RD E2 接続用の HiRDB External Data Access Adapter の場合：512
- ORACLE 接続用の HiRDB External Data Access Adapter の場合：2048

Ｆ：pd_assurance_table_no オペランドの値

Ｇ：サーバ内の最大トランザクション数（2 × e ＋ 7）

Ｈ：pd_dfw_awt_process オペランドの値

Ｉ：フロントエンドサーバで指定した pdplgprm オペランドの総数

Ji：フロントエンドサーバで指定した i 番目の pdplgprm オペランドで指定した共用メモリのサイズ

K：ディクショナリサーバで指定した pdplgprm オペランドの総数

Li：ディクショナリサーバで指定した i 番目の pdplgprm オペランドで指定した共用メモリのサイズ

M：バックエンドサーバで指定した pdplgprm オペランドの総数

Ni：バックエンドサーバで指定した i 番目の pdplgprm オペランドで指定した共用メモリのサイズ

Ｐ：pd_max_commit_write_reclaim_no オペランドの値

Q：pd_max_resident_rdarea_no オペランドの値

R：pd_max_resident_rdarea_shm_no オペランドの値

注※ 1

ディクショナリサーバの場合は pd_max_dic_process オペランドの値となります。バックエンドサーバの場合は pd_max_bes_process オペランドの値となります。ただし，pd_max_dic_process 又は pd_max_bes_process オペランドを省略する場合は，pd_max_users オペランドの値となります。

注※ 2

DML で使用するプラグイン関数の総数及び DML で使用するプラグイン関数のパラメタの総数は，次に示す SQL で求められます。

```
SELECT COUNT(*),SUM(N_PARAM) FROM MASTER.SQL_PLUGIN_ROUTINES
WHERE PLUGIN_NAME = 'プラグイン名称'
AND (TIMING_DESCRIPTOR = 'ADT_FUNCTION'
    OR TIMING_DESCRIPTOR IS NULL
    OR TIMING_DESCRIPTOR = 'BEFORE_INSERT'
    OR TIMING_DESCRIPTOR = 'AFTER_INSERT'
    OR TIMING_DESCRIPTOR = 'BEFORE_UPDATE'
    OR TIMING_DESCRIPTOR = 'AFTER_UPDATE'
    OR TIMING_DESCRIPTOR = 'BEFORE_DELETE'
    OR TIMING_DESCRIPTOR = 'AFTER_DELETE'
    OR TIMING_DESCRIPTOR = 'BEFORE_PURGE_TABLE'
    OR TIMING_DESCRIPTOR = 'AFTER_PURGE_TABLE'
    OR TIMING_DESCRIPTOR = 'INDEX_SEARCH'
    OR TIMING_DESCRIPTOR = 'INDEX_COUNT'
    OR TIMING_DESCRIPTOR = 'INDEX_INSERT'
    OR TIMING_DESCRIPTOR = 'INDEX_BEFORE_UPDATE'
    OR TIMING_DESCRIPTOR = 'INDEX_AFTER_UPDATE'
    OR TIMING_DESCRIPTOR = 'INDEX_DELETE'
    OR TIMING_DESCRIPTOR = 'PURGE_INDEX'
    OR TIMING_DESCRIPTOR = 'INDEX_MAINTENANCE_DEFERRED'
    OR TIMING_DESCRIPTOR = 'BEFORE_INSERT_DC'
    OR TIMING_DESCRIPTOR = 'BEFORE_UPDATE_DC'
    OR TIMING_DESCRIPTOR = 'BEFORE_DATA_CHECK'
    OR TIMING_DESCRIPTOR = 'AFTER_DATA_CHECK')
```

## 15.2.5 グローバルバッファが使用する共用メモリの計算式

### (1) 影響分散スタンバイレス型系切り替え機能を使用しない場合

グローバルバッファが使用する共用メモリサイズは，ディクショナリサーバ又はバックエンドサーバごとに計算式 1 に代入して求めます。サーバマシンごとに算出する場合，pdbuffer 文のオプションの指定によって計算対象になるかどうかが異なります。pdbuffer 文のオプションによる計算条件を次の表に示します。

表 15-9 pdbuffer 文のオプションによる計算条件（影響分散スタンバイレス型系切り替え機能を使用しない場合）

| pdbuffer 文のオプション | 計算条件 |
|---|---|
| -r | -r で指定した RD エリアがあるサーバの場合，計算対象とします。 |
| -i | -i で指定したインデクスが格納されている RD エリアがあるサーバの場合，計算対象とします。 |
| -b | -b で指定した RD エリアがあるサーバの場合，計算対象とします。 |
| -o | そのサーバにある全 RD エリア中で pdbuffer -r で指定していない RD エリアがある場合，計算対象とします。 |

pd_dbbuff_modify オペランドに Y を指定している場合は，計算式 2 を加算します。計算式 1 及び 2 で求めた値を合計した値が，そのサーバでグローバルバッファが使用する共用メモリの所要量です。

なお，pdbuffer オペランドを省略した場合，HiRDB が共用メモリサイズを自動計算するので見積もりは必要ありません。

### (a) グローバルバッファが使用する共用メモリを実メモリ上に固定しない場合

グローバルバッファが使用する共用メモリを実メモリ上に固定しない場合（pd_dbbuff_attribute オペランドに free を指定した場合）は，次の計算式で求めます。

| 計算式の種類 | 共用メモリの計算式（単位：キロバイト） |
|---|---|
| 計算式 1 | ●32 ビットモードの場合<br>$\sum_{i=1}^{n}$ {《管理領域部》＋《データ格納部》} ÷ 1024<br>《管理領域部》：<br>↑ {736 ＋ 64 ＋（296 ＋ 64※1）×（Pi ＋ 4）<br>＋（124 ＋ 80※2 ＋ 96 × A × Mi）× Ui} ÷ 4096 ↑ × 4096<br>《データ格納部》：<br>Si × {Pi ＋ 4 ＋（Ui × Mi × A)}<br>●64 ビットモードの場合<br>$\sum_{i=1}^{n}$ {《管理領域部》＋《データ格納部》} ÷ 1024<br>《管理領域部》：<br>↑ {928 ＋ 64 ＋（480 ＋ 112※1）×（Pi ＋ 4）<br>＋（176 ＋ 96※2 ＋ 136 × A × Mi）× Ui} ÷ 4096 ↑ × 4096<br>《データ格納部》：<br>Si × {Pi ＋ 4 ＋（Ui × Mi × A)} |
| 計算式 2 | ●32 ビットモードの場合<br>{↑［((s × 1024 ÷ 4）÷ 8）＋ 112］÷ 4096 ↑ × 4096 × ↑ a ÷（s × 1024）↑<br>}<br>●64 ビットモードの場合<br>{↑［((s × 1024 ÷ 4）÷ 8）＋ 144］÷ 4096 ↑ × 4096 × ↑ a ÷（s × 1024）↑<br>} |

n：グローバルバッファプール数

i：計算対象のグローバルバッファプール定義

P：グローバルバッファ面数

A：非同期 READ 機能を使用する場合は 2，使用しない場合は 1

M：一括入力最大ページ数
pd_max_ard_process オペランドに 1 以上を指定した場合は指定値の 2 倍です。

U：同時実行最大プリフェッチ数

S：グローバルバッファに割り当てた RD エリアの最大ページ長

s：SHMMAX 指定値

a：計算式 1 の総計

注※ 1　LOB 用グローバルバッファの場合に加算します。

注※ 2　pd_max_ard_process オペランドに 1 以上を指定した場合に加算します。

### (b) グローバルバッファが使用する共用メモリを実メモリ上に固定する場合

グローバルバッファが使用する共用メモリを実メモリ上に固定する場合（pd_dbbuff_attribute オペランドに fixed を指定した場合）は，次の計算式で求めます。

| 計算式の種類 | 共用メモリの計算式（単位：キロバイト） |
|---|---|
| 計算式 1 | **● 32 ビットモードの場合**<br>$\sum_{i=1}^{n}$ {↑ {《管理領域部》＋《データ格納部》} ÷ p ↑× p} ÷ 1024<br>《管理領域部》：<br>↑ {↑ {736 ＋ 64 ＋（296 ＋ 64※1）×（Pi ＋ 4）<br>＋（124 ＋ 80※2 ＋ 96 × A × Mi）× Ui} ÷ 4096 ↑× 4096<br>《データ格納部》：<br>Si × {Pi ＋ 4 ＋（Ui × Mi × A)}<br>**● 64 ビットモードの場合**<br>$\sum_{i=1}^{n}$ {↑ {《管理領域部》＋《データ格納部》} ÷ p ↑× p} ÷ 1024<br>《管理領域部》：<br>↑ {↑ {928 ＋ 64 ＋（480 ＋ 112※1）×（Pi ＋ 4）<br>＋（176 ＋ 96※2 ＋ 136 × A × Mi）× Ui} ÷ 4096 ↑× 4096<br>《データ格納部》：<br>Si × {Pi ＋ 4 ＋（Ui × Mi × A)} |
| 計算式 2 | **● 32 ビットモードの場合**<br>{↑ {↑ [((s × 1024 ÷ 4）÷ 8）＋ 112] ÷ 4096 ↑× 4096 ×↑ a ÷（s × 1024）↑} ÷ p ↑× p}<br>**● 64 ビットモードの場合**<br>{↑ {↑ [((s × 1024 ÷ 4）÷ 8）＋ 144] ÷ 4096 ↑× 4096 ×↑ a ÷（s × 1024）↑} ÷ p ↑× p} |

n：グローバルバッファプール数

i：計算対象のグローバルバッファプール定義

P：グローバルバッファ面数

A：非同期 READ 機能を使用する場合は 2，使用しない場合は 1

M：一括入力最大ページ数
pd_max_ard_process オペランドに 1 以上を指定した場合は指定値の 2 倍です。

U：同時実行最大プリフェッチ数

S：グローバルバッファに割り当てた RD エリアの最大ページ長

p：Windows の Large Page でのページサイズ
pdntenv コマンドで確認できます。

s：SHMMAX 指定値

a：計算式 1 の総計

注※ 1　LOB 用グローバルバッファの場合に加算します。

注※ 2　pd_max_ard_process オペランドに 1 以上を指定した場合に加算します。

## (2) 影響分散スタンバイレス型系切り替え機能を使用する場合

影響分散スタンバイレス型系切り替え機能を使用する場合，ユニットごとに計算式に代入して求めます。ユニットごとに算出する場合，pdbuffer 文のオプションの指定によって計算対象になるかどうかが異なります。pdbuffer 文のオプションによる計算条件を次の表に示します。

**表 15-10　pdbuffer 文のオプションによる計算条件（影響分散スタンバイレス型系切り替え機能を使用する場合）**

| pdbuffer 文のオプション | 計算条件 |
|---|---|
| -r，-b | -r で指定した RD エリアで，同じ HA グループに属する RD エリアの場合，計算対象とします。 |
| -i | -i で指定したインデクスが格納されている RD エリアが，同じ HA グループに属する RD エリアの場合，計算対象とします。 |
| -o | 同じ HA グループ内で，pdbuffer -r オプションで指定していない RD エリアがある場合，計算対象とします。 |

なお，pdbuffer オペランドを省略した場合，HiRDB が共用メモリサイズを自動計算するので見積もりは必要ありません。

### (a) グローバルバッファが使用する共用メモリを実メモリ上に固定しない場合

グローバルバッファが使用する共用メモリを実メモリ上に固定しない場合（pd_dbbuff_attribute オペランドに free を指定した場合）は，次の計算式で求めます。

| 共用メモリの計算式（単位：キロバイト） |
|---|

**● 32 ビットモードの場合**

n
Σ {《管理領域部》＋《データ格納部》} ÷ 1024
i=1
《管理領域部》：
((96 ＋ ((736 × (A ＋ B)) ＋ (288 × (F ＋ (8 × (A ＋ B))))
＋ (8 × F × (A ＋ B))) ＋ 16) ＋ H ＋ D)
《データ格納部》：
2048 ＋ G ＋ (E × F ＋ (8 × (A ＋ B)))

**● 64 ビットモードの場合**

n
Σ {《管理領域部》＋《データ格納部》} ÷ 1024
i=1
《管理領域部》：
((144 ＋ ((928 × (A ＋ B)) ＋ (464 × (F ＋ (8 × (A ＋ B))))
＋ (16 × F × (A ＋ B)))) +16 ＋ H ＋ D)
《データ格納部》：
2048 ＋ G ＋ (E × F ＋ (8 × (A ＋ B)))

n：このユニットに割り当てられるグローバルバッファプール数

i：計算対象のグローバルバッファプール定義

A：ホスト BES 数

B：受け入れ可能なゲスト BES の最大数

C：一括入力ページ数（pdbuffer -p に指定した値）

D：プリフェッチ機能使用時（pdbuffer -m 指定）に加算します。
32 ビットモードの場合
2 × (((80 × U × C) ＋ (80 × U) ＋ (124 × U) ＋ (8 × U × C)) × (A ＋ B))
64 ビットモードの場合
2 × (((112 × U × C) ＋ (96 × U) ＋ (176 × U) ＋ (16 × U × C)) × (A ＋ B))

E：pdbuffer 文のオプションの指定によって値が異なります。オプションと計算式を次に示します。

| pdbuffer 文のオプション | 最大値の計算式 |
|---|---|
| -r，-b | (MAX（バッファサイズ (pdbuffer -l の値)，MAX（指定した RD エリアで同じ HA グループに属する RD エリアのページサイズ))) |
| -i | (MAX（バッファサイズ（pdbuffer - の値)，MAX（-i で指定したインデクスが格納されている RD エリアが同じ HA グループに属する RD エリアのページサイズ))) |
| -o | (MAX（バッファサイズ（pdbuffer -l に指定した値)，MAX（同じ HA グループ内で pdbuffer -r オプションで指定していない RD エリアのページサイズ))) |

F：バッファ面数（pdbuffer -n の値）

G：プリフェッチ機能使用時（pdbuffer -m 指定）に加算します。
2 × ((E × U × C) × (A ＋ B))

H：LOB 用 RD エリアが指定されている（pdbuffer -b 指定）場合に加算します。
32 ビットモードの場合：64 × (F ＋ (8 × (A ＋ B)))

64 ビットモードの場合：112 ×（F ＋（8 ×（A ＋ B）））

U：同時実行最大プリフェッチ数（pdbuffer -m の値）

### (b) グローバルバッファが使用する共用メモリを実メモリ上に固定する場合

グローバルバッファが使用する共用メモリを実メモリ上に固定する場合（pd_dbbuff_attribute オペランドに fixed を指定した場合）は，次の計算式で求めます。

**共用メモリの計算式（単位：キロバイト）**

**● 32 ビットモードの場合**

$$\sum_{i=1}^{n} \{\uparrow \{《\text{管理領域部}》 + 《\text{データ格納部}》\} \div p \uparrow \times p\} \div 1024$$

《管理領域部》：
((96 ＋ ((736 ×（A ＋ B）) ＋（288 ×（F ＋（8 ×（A ＋ B））))
＋（8 × F ×（A ＋ B））＋ 16）＋ H ＋ D）

《データ格納部》：
＋ 2048 ＋ G ＋（E × F ＋（8 ×（A ＋ B）））

**● 64 ビットモードの場合**

$$\sum_{i=1}^{n} \{\uparrow \{《\text{管理領域部}》 + 《\text{データ格納部}》\} \div p \uparrow \times p\} \div 1024$$

《管理領域部》：
((144 ＋ ((928 ×（A ＋ B）) ＋（464 ×（F ＋（8 ×（A ＋ B））))
＋（16 × F ×（A ＋ B））+16）＋ H ＋ D）

《データ格納部》：
2048 ＋ G ＋（E × F ＋（8 ×（A ＋ B）））

n：このユニットに割り当てられるグローバルバッファプール数

i：計算対象のグローバルバッファプール定義

A：ホスト BES 数

B：受け入れ可能なゲスト BES の最大数

C：一括入力ページ数（pdbuffer -p に指定した値）

D：プリフェッチ機能使用時（pdbuffer -m 指定）に加算します。
32 ビットモードの場合
2 ×（((80 × U × C）＋（80 × U）＋（124 × U）＋（8 × U × C））×（A ＋ B））
64 ビットモードの場合
2 ×（((112 × U × C）＋（96 × U）＋（176 × U）＋（16 × U × C））×（A ＋ B））

E：pdbuffer 文のオプションの指定によって値が異なります。オプションと計算式を次に示します。

| pdbuffer 文のオプション | 最大値の計算式 |
|---|---|
| -r，-b | （MAX（バッファサイズ (pdbuffer -l の値)，MAX（指定した RD エリアで同じ HA グループに属する RD エリアのページサイズ））） |
| -i | （MAX（バッファサイズ（pdbuffer - の値)，MAX（-i で指定したインデクスが格納されている RD エリアが同じ HA グループに属する RD エリアのページサイズ））） |
| -o | （MAX（バッファサイズ（pdbuffer -l に指定した値)，MAX（同じ HA グループ内で pdbuffer -r オプションで指定していない RD エリアのページサイズ））） |

F：バッファ面数（pdbuffer -n の値）

G：プリフェッチ機能使用時（pdbuffer -m 指定）に加算します。
2 ×（(E × U × C) ×（A ＋ B)）

H：LOB 用 RD エリアが指定されている（pdbuffer -b 指定）場合に加算します。
32 ビットモードの場合：64 ×（F ＋（8 ×（A ＋ B)))
64 ビットモードの場合：112 ×（F ＋（8 ×（A ＋ B)))

U：同時実行最大プリフェッチ数（pdbuffer -m の値）

p：Windows の Large Page でのページサイズ
pdntenv コマンドで確認できます。

## 15.2.6 SQL 実行時に必要なメモリ所要量の計算式

### (1) グループ分け高速化機能実行時に必要なメモリ所要量の求め方

クライアント環境定義で PDSQLOPTLVL オペランドを指定するか，HiRDB システム定義で pd_optimize_level オペランドを指定（又は省略）した場合，適用条件を満たす SQL を実行すると，グループ分け高速化機能が働きます。このとき，HiRDB はクライアント環境定義の PDAGGR オペランドの値に基づいてプロセス固有メモリを確保します。確保するメモリサイズを次に示します。

グループ分け高速化機能実行時に必要なメモリ所要量は，バックエンドサーバを定義するサーバマシンについてだけ計算してください。

**計算式**

```
e ＋↑ d ÷ 4 ↑× 4 ＋↑（17 ＋ 4 × a ＋ 4 × b ＋ c ＋ d）÷ 4 ↑× 4 ×（N ＋ 1）
                                                        （単位：バイト）
```

a：グループ化する列の数

b：集合関数の演算数
COUNT，SUM，MAX，MIN は一つにつき 1 で換算します。
AVG（COUNT），AVG（SUM）は一つにつき 2 で換算します。

c：グループ化する列の列長（表 15-11 を参照して求めてください）

d：集合関数の演算領域長（表 15-11 を参照して求めてください）

e：32 ビットモードの場合：MAX（4 × N × 24 , 16408）
64 ビットモードの場合：MAX（8 × N × 40 , 32808）

N：クライアント環境定義の PDAGGR オペランドの値

**表 15-11 グループ化するときの列の長さ及び集合関数の演算領域の長さ**

| 列のデータ型 | グループ化する列の列長 | 集合関数の演算領域長※1 |
|---|---|---|
| INTEGER | 4 | 6 |
| SMALLINT | 2 | 4※2 |
| DECIMAL(p,s) | ↑（p ＋ 1）÷ 2 ↑ | ↑（p ＋ 7）÷ 2 ↑※3 |

| 列のデータ型 | グループ化する列の列長 | 集合関数の演算領域長※1 |
|---|---|---|
| FLOAT | 8 | 10 |
| SMALLFLT | 4 | 6 |
| INTERVAL YEAR TO DAY | 5 | 8 |
| INTERVAL HOUR TO SECOND | 4 | 6 |
| CHAR(n) | n | n＋3 |
| VARCHAR(n) | n＋2 | n＋5 |
| NCHAR(n) | 2×n | 2×n＋2 |
| NVARCHAR(n) | 2×n＋2 | 2×n＋4 |
| MCHAR(n) | n | n＋3 |
| MVARCHAR(n) | n＋2 | n＋5 |
| DATE | 4 | 6 |
| TIME | 3 | 6 |
| BLOB(n) | － | － |
| BINARY(n) | n＋2 | n＋5 |

（凡例）－：該当しません。

注※1
集合関数が COUNT の場合，集合関数演算領域長はデータ型にかかわらず 6 になります。

注※2
集合関数が AVG，SUM の場合，集合関数演算領域長は 6 になります。

注※3
集合関数が AVG，SUM の場合，集合関数演算領域長は次の値になります。
集合関数の値の型が DECIMAL 型で精度が 29 けたのとき：18
集合関数の値の型が DECIMAL 型で精度が 38 けたのとき：23
集合関数の値のデータ型の規則については，マニュアル「HiRDB Version 8 SQL リファレンス」の「集合関数」を参照してください。

## (2) 列ごとのデータ抑制指定時に必要なメモリ所要量の求め方

列ごとのデータ抑制を指定（CREATE TABLE の列定義に SUPPRESS を指定）した表に対してアクセスするときに使用するメモリサイズは，次に示す計算式で求められます。

**計算式**

a＋128　（単位：バイト）

a：表中の列ごとのデータ抑制が指定されている列の定義長の合計値

## (3) ハッシュジョイン及び副問合せのハッシュ実行時に必要なメモリ所要量の求め方

クライアント環境定義で PDADDITIONALOPTLVL オペランドを指定するか，HiRDB システム定義で pd_additional_optimize_level オペランドを指定すると，SQL 拡張最適化オプションが働きます。この SQL 拡張最適化オプションで，「ハッシュジョイン，副問合せのハッシュ実行の適用

（APPLY_HASH_JOIN)」を指定した場合，表の結合又は副問合せの SQL を実行すると，次に示すメモリサイズのプロセス固有領域を確保します。

**計算式**

**● 32 ビットモードの場合**

$$\sum_{i=1}^{a}(13 \times 1024 + 6 \times 1024 \times b + c)$$

**● 64 ビットモードの場合**

$$\sum_{i=1}^{a}(13 \times 1024 + 7 \times 1024 \times b + c)$$

（単位：バイト）

a：SELECT 文のハッシュジョイン最大数
SELECT 文のハッシュジョイン最大数については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

b：ハッシュ表行数によって適用されるハッシュジョイン処理を求めて，次に示す表から代入する値を決定してください。

| ハッシュ表行数の目安 | 適用されるハッシュジョイン処理 | | b の値 |
|---|---|---|---|
| 1500 以内 | 一括ハッシュジョイン | | 0.5 |
| 1500 ×（バケット分割数÷ 3）以内 | バケット分割ハッシュジョイン | 1 レベルバケット分割 | 1 |
| 1500 ×（バケット分割数÷ 3）$^2$ 以内 | | 2 レベルバケット分割 | 2 |
| 1500 ×（バケット分割数÷ 3）$^2$ を超える場合 | | 3 レベルバケット分割 | 3 |

**ハッシュ表行数：**ジョインの場合はジョインの内表件数です。副問合せの場合は探索条件中の外への参照列を含む述語を除いた副問合せ探索件数です。

**バケット分割数：**MIN｛↓（ハッシュ表サイズ÷ 2）÷ハッシュ表ページ長↓，64｝

**ハッシュ表サイズ：**HiRDB システム定義の pd_hash_table_size オペランド，又はクライアント環境変数の PDHASHTBLSIZE オペランドで指定した値です。

**ハッシュ表ページ長：**次に示す表から c（ハッシュ表最大行長）に対応するハッシュ表ページ長を選択してください。

| ハッシュ表最大行長 | ハッシュ表ページ長（単位：バイト） |
|---|---|
| 0 ～ 1012 | 4096 |
| 1013 ～ 2036 | 8192 |
| 2037 ～ 4084 | 16384 |
| 4085 ～ 16360 | 32768 |
| 16361 ～ 32720 | ↑（ハッシュ表最大行長＋ 48）÷ 2048 ↑× 2048 |

c：ハッシュ表最大行長
ハッシュ表最大行長については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。

## (4) 外部表検索時に必要なメモリ所要量の求め方

外部表検索時（HiRDB External Data Access 機能使用時）に必要なメモリ所要量の計算式を次に示します。アクセスパス表示ユティリティ（pdvwopt コマンド）で外部表を検索する SQL 文を確認できます。なお，外部表に対する SQL 実行時に使用するメモリの最大サイズは 2,147,483,647 バイトとなります。

**計算式**

```
a ＋ 16 ＋ 16 × b ＋ c
外部表を検索する SQL 文中に埋込み変数又は？パラメタがある場合に加算します。
＋ 16 ＋ 16 × e ＋ d
                                                （単位：バイト）
```

ａ：外部表を検索する SQL 文の長さ

ｂ：検索対象となる外部表の列数

ｃ：＜検索対象の外部サーバの，外部サーバ情報定義に pd_hb_ary_fec_num オペランドがある場合＞

- pd_hb_ary_fec_num オペランドを指定しているとき
  検索対象となる外部表の列長の総和（表 15-5 を参照）× pd_hb_ary_fec_num オペランドの値
- pd_hb_ary_fec_num オペランドを指定していないとき
  MAX｛検索対象となる外部表の列長の総和（表 15-5 を参照），pd_sql_send_buff_size の値× 2048｝

＜検索対象の外部サーバの，外部サーバ情報定義に pd_hb_ary_fec_num オペランドがない場合＞
外部表の列長の総和（表 15-5 を参照を参照）

ｄ：外部表を検索する SQL 文中の埋込み変数又は？パラメタの列長の総和（表 15-5 を参照）

ｅ：外部表を検索する SQL 文中の埋込み変数又は？パラメタ数

## (5) スナップショット方式指定時に必要なメモリ所要量の求め方

pd_pageaccess_mode オペランドを省略した場合又は SNAPSHOT を指定した場合，スナップショット方式を適用する SQL 文を実行すると，データベース検索時のページアクセス方式にスナップショット方式を使用します。このとき，表又はインデクスの格納 RD エリアのページサイズに基づいて，動的に次に示すメモリサイズのプロセス固有領域を確保します。

**計算式**

```
a × 2　（単位：バイト）
```

ａ：検索対象の表又はインデクスが格納されている RD エリア中の最大ページ長。
ただし，LOB 用 RD エリアは除きます。

## (6) 先頭から n 行の検索結果を取得する機能実行時に必要なメモリ所要量の求め方

先頭から n 行の検索結果を取得する機能を使用すると，検索結果の先頭（又はユーザが指定した先頭からのオフセット行数分読み飛ばした位置）から n 行取得できます。

LIMIT 句に指定した行数が 1 以上で，（オフセット行数＋ LIMIT 句に指定した行数）の値が 32,767 以下の場合，（オフセット行数＋ LIMIT 句に指定した行数）以内に入り得る行をメモリに保持します。確保するプロセス固有領域のメモリサイズは，次に示す計算式で求められます。なお，（オフセット行数＋

LIMIT 句に指定した行数）の値が 32,768 以上になる場合は作業表を作成するため，「18. 作業表用ファイルの容量の見積もり」を参照してください。

**計算式**

{100 ＋（a ＋ 2）×（オフセット行数＋ LIMIT 句に指定した行数)} × b　（単位：バイト）

a：行長

行長は 32,720 バイト以下でなければなりません。行長は次の計算式で求められます。

$$\sum_{i=1}^{m}(Ai) + 2 \times m + 4 + c$$

（単位：バイト）

m：選択式，GROUP BY 句，又は ORDER BY 句に指定した列数

FOR UPDATE 句を指定した場合は 1 を加算してください。ただし，選択式に ROW を指定している場合は表の全列数になります。

Ai：先頭 n 行保持領域に格納する行の i 番目の列データ長

列のデータ長については，「表 16-1　データ長一覧」を参照し，d に定義長を代入して求めてください。

ただし，BLOB データ，定義長が 256 バイト以上の文字データ（各国・混在文字データも含む），BINARY データのうち，下記に属さない列の場合は 12 バイトになります。

- DISTINCT 句指定の選択式に指定する列
- UNION [ALL] によって集合演算対象となっている問合せ指定中の選択式
- ORDER BY 句に指定した列

また，FOR UPDATE 句を指定した場合に，m に加算した 1 に対応する Ai は 12 バイトとします。

c：8

ただし，次の場合は 0 になります。

- 検索対象の表に EX モードで排他が掛かっている場合
- WITHOUT LOCK を指定した場合
- グループ分け高速化機能を指定した場合
- 複数の表を結合する場合

b：先頭 n 行保持領域数

先頭 n 行保持領域数は次の計算式で求められます。

1 ＋ UNION [ALL] 句指定数

## （7）探索条件にインデクス型プラグイン専用関数を指定した SQL 文実行時に必要なメモリ所要量の求め方

探索条件にインデクス型プラグイン専用関数を指定した SQL 文の実行時に確保するプロセス固有領域のメモリサイズは，次に示す計算式で求められます。

**計算式**

a × 500 ＋（20 ＋ 6）× 800 ＋ 16　（単位：バイト）

a：行長。行長は次の計算式で求められます。

$$\sum_{i=1}^{m}(Ai) + 4 \times (m + 2) + 12 + 4 + 8$$

（単位：バイト）

m：選択式，結合条件，GROUP BY 句，又は ORDER BY 句に指定した列数
FOR UPDATE 句を指定した場合は 1 を加算してください。ただし，選択式に ROW を指定している場合は表の全列数になります。

Ai：取り出す行の i 番目の列データ長
列のデータ長については，「表 16-1　データ長一覧」を参照し，d に定義長を代入して求めてください。
ただし，BLOB データ，又は定義長が 256 バイト以上の文字データ（各国・混在文字データも含む）で，下記に属さない列の場合は 12 バイトになります。

- 結合条件中に指定する列（結合列）
- DISTINCT 句指定の選択式に指定する列
- 限定述語の副問合せ中の選択式に指定する列
- IN 述語の副問合せ中の選択式に指定する列
- UNION [ALL]，又は EXCEPT [ALL] によって集合演算対象となっている問合せ指定中の選択式
- ORDER BY 句に指定した列

また，FOR UPDATE 句を指定した場合に，m に加算した 1 に対応する Ai は 12 バイトとします。

## （8）拡張 SQL エラー情報出力機能使用時に必要なメモリ所要量の求め方

拡張 SQL エラー情報出力機能を使用した場合，次のときにプロセス固有領域を確保します。

### （a）OPEN 文実行時

**計算式**

**● 32 ビットモードの場合**
（16 ＋ 16 × m）＋ a
**● 64 ビットモードの場合**
（16 ＋ 24 × m）＋ a

（単位：バイト）

a：？パラメタ又は埋込み変数のデータ長の合計

$$a = \sum_{i=1}^{m}(ai)$$

m：SQL 文中の？パラメタ又は埋込み変数の数

ai：i 番目の？パラメタ又は埋込み変数のデータ長
データ長については，表 15-5 を参照して算出します。

#### (b) 定義系 SQL の PREPARE 文実行時

**計算式**

SQL 文長 +20　（単位：バイト）

### (9) 部分構造インデクスの定義，又は部分構造インデクスを定義した表の更新時に必要なメモリ所要量の求め方

#### (a) 部分構造インデクスの定義時

定義系 SQL の CREATE INDEX で部分構造インデクスを定義する場合に確保するプロセス固有領域は，次に示す計算式で求められます。

**計算式**

（インデクスキー長※× 100 ＋ 64）　（単位：バイト）

注※
表に定義する部分構造インデクスの最大定義長です。

#### (b) 部分構造インデクスを定義した表の更新時

操作系 SQL の INSERT，UPDATE，又は DELETE で部分構造インデクスを定義した表を更新する場合に確保するプロセス固有領域は，次に示す計算式で求められます。

**計算式**

（インデクスキー長※1× 100 ＋ 64 ＋ 128）＋ Σ（インデクスキー長＋ 128）※2　（単位：バイト）

注※ 1
表に定義している部分構造インデクスの最大定義長です。

注※ 2
USING UNIQUE TAG 指定の部分構造インデクス数です。

## 15.2.7 SQL 前処理時に必要なメモリ所要量の計算式

### (1) ストアドプロシジャを使用しない場合に必要なメモリ所要量の求め方

ストアドプロシジャを使用しない場合，SQL 前処理時に確保するメモリサイズは，次に示す計算式で求められます。

**計算式**

```
↑ {
   (2539 + Si × 70 + Pi × 20 + Ti × 980 + Ci × 68 + Wi × 818 + Ki × 416 + Li × 5
   + Di × 116 + Ari × 108 + Gi × 44 + Ori × 10 + Sli × 40 + Upi × 96 + Fi × 90
   + Ti × Cwi × 48 + MAX（Pi，Wpi）× 52 + MAX（Ti，Sli − 1）× 96
   + MAX（Ti × 2，Wi）× 24 + MAX（Ti × 3，Wi）× 24
   + MAX {MAX（Ti，Ori + Gi + Si + Fi），Sli − 1} × 24
 } × 1.2 ÷ 1024 ↑× CLS
                                                    （単位：キロバイト）
```

Si：SQL 文中の検索項目数

Pi：SQL 文中の埋込み変数，？パラメタ又は SQL パラメタの数

Ti：SQL 文中の表名の数

Ci：SQL 文中の列名の数

Wi：SQL 文中の論理演算子（AND 及び OR）に出てくる述語の数

Ki：SQL 文中の定数の数

Li：SQL 文中の定数の長さの合計（単位：バイト）

Di：SQL 文中に定義された格納条件の総数

Ari：SQL 文中の四則演算及び連結演算の数

Gi：SQL 文中の GROUP BY 句に指定した列の数

Ori：SQL 文中の ORDER BY 句に指定した列指定又はソート項目指定番号の数

Fi：SQL 文中の集合関数及びスカラ関数の総数

Sli：SQL 文中の問合せ指定の数

Upi：SQL 文中の更新列数

Cwi：SQL 文中の CASE 式中の WHEN の数

Wpi：SQL 文中の WITH 句に対応する変数の数

CLS：SQL オブジェクト内の一つのアクセスパスが生成される領域の数※

注※

SQL オブジェクト内の一つのアクセスパスが生成される領域の数は，次に示す計算式で求められます。

**計算式**

**SELECT_APSL が適用されている場合※**

```
a + b × 4 + c + d + e × 2
```

**SELECT_APSL が適用されていない場合※**

```
a + b + c + d + e
```

a：フロントエンドサーバ数

フロントエンドサーバ数は 1 を指定します。

b：表数

表数は次に示す計算式で求めます。
実表数＋相関名数

c：集合演算サーバ数
集合関数の指定がある場合は 1 を，指定がない場合は 0 を指定します。

d：GROUP BY 句，DISTINCT 又は ORDER BY 句がある問合せ指定の数

e：ジョインサーバ数
ジョインサーバ数は次に示す計算式で求めます。
b － SQL 文中の問合せ指定の数

注※
SELECT_APSL が適用されているかどうかについては，アクセスパス表示ユティリティ（pdvwopt）を使用すると分かります。アクセスパス表示ユティリティ（pdvwopt）については，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

### (2) ストアドプロシジャを使用する場合に必要なメモリ所要量の求め方

ストアドプロシジャを使用する場合，SQL 前処理時に確保するメモリサイズ（単位：キロバイト）は，「(1) ストアドプロシジャを使用しない場合に必要なメモリ所要量の求め方」の計算式で求めた値に，ストアドプロシジャごとのプロシジャ制御用オブジェクト長を加算します。プロシジャ制御用オブジェクト長の計算式については，システム共通定義の pd_sql_object_cache_size オペランドの 1 ストアドプロシジャのプロシジャ制御用オブジェクト長を参照してください。1 ストアドプロシジャのプロシジャ制御用オブジェクト長については，マニュアル「HiRDB Version 8 システム定義」の「1 ルーチンのルーチン制御用オブジェクト長の計算式」を参照してください。

## 15.2.8 BLOB 型データの検索又は更新時に必要なメモリ所要量の計算式（フロントエンドサーバの場合）

BLOB 型データの検索又は更新時に必要なメモリ所要量は次に示す計算式で求められます。

**計算式**

```
a ＋ b ＋ 7　　（単位：キロバイト）
```

a：1SQL 文中に指定する BLOB 型入力変数又は出力変数で，実行する SQL 文の中で次に示す計算式の結果が最大となる値です。

$$\uparrow \left\{ \sum_{i=1}^{c} (\text{BLOB 型入力変数 i の実長}^{※1} \times 2 + 58) + \sum_{j=1}^{d} (\text{BLOB 型出力変数 j の定義長}^{※2} + 26) \right\} \div 1024 \uparrow$$

注※ 1
埋込み変数で UAP から HiRDB サーバに受け渡された BLOB 型データの実際の長さです。

注※ 2

HiRDB から UAP に返却する BLOB 型データを受け取る UAP の埋込み変数の宣言長です。

b：同時オープン中のカーソルで結合検索を行う SQL 文の組み合わせで，次に示す計算式の結果が最大となる値です。

256 ×同時オープン中のカーソル数

c：入力変数の数

d：出力変数の数

## 15.2.9 BLOB 型データの検索又は更新時に必要なメモリ所要量の計算式（バックエンドサーバ又はディクショナリサーバの場合）

BLOB 型データの検索又は更新時に必要なメモリ所要量は次に示す計算式で求められます。

**計算式**

```
a ＋ b
                                                （単位：キロバイト）
```

a：1SQL 文中に指定する BLOB 型入力変数又は出力変数について，実行する SQL 文の中で，次に示す計算式の結果が最大となる値です。

```
↑ {
     c
    Σ（BLOB 型入力変数 i の実長※＋ 118 ＋ 70 ×出力変数の数）
   i=1
} ÷ 1024 ↑
```

注※

埋込み変数で UAP から HiRDB サーバに受け渡された BLOB 型データの実際の長さです。

b：同時オープン中のカーソルで BLOB 型データの検索を行う SQL 文の組み合わせで，次に示す計算式の結果が最大となる値です。

```
d
Σ {280 ＋ 184 ×（SQLi に記述した表数＋ 1)}
i=1
```

c：入力変数の数

d：カーソル数

## 15.2.10 ブロック転送又は配列 FETCH で必要なメモリ所要量の計算式（フロントエンドサーバの場合）

ブロック転送又は配列 FETCH で必要なメモリ所要量は，次の計算式で求められます。

| 条件 | PDBLKBUFFSIZE オペランドの指定値 | |
|---|---|---|
| | 省略又は 0 | 1 以上 |
| FETCH 文の INTO 句に配列型の埋込み変数を指定する | 計算式 1 | |

| 条件 | | PDBLKBUFFSIZE オペランドの指定値 | |
|---|---|---|---|
| | | 省略又は 0 | 1 以上 |
| FETCH 文の INTO 句に配列型の埋込み変数を指定しない | PDBLKF オペランドを省略又は 1 | － | 計算式 2 |
| | PDBLKF オペランドが 2 以上 | 計算式 1 | |

（凡例）－：該当しません。

**計算式 1**

↑｛864 ＋ 16 × a ＋（6 × a ＋ 2 × d ＋ b）× c｝÷ 1024 ↑
（単位：キロバイト）

a：SELECT 句で指定する検索項目数

b：FETCH 文で受け取る検索結果中の 1 行のデータ長（各列の最大長の合計。単位はバイト）

c：PDBLKF オペランドの指定値又は配列数

d：SELECT 句で指定する検索項目で，BINARY 型を指定した選択式の数

**計算式 2**

MAX（$X_1$，$X_2$）
（単位：キロバイト）

$X_1$：↑（864 ＋ 22 × a ＋ 2 × c ＋ b）÷ 1024 ↑

$X_2$：PDBLKBUFFSIZE オペランドの値

a：SELECT 句で指定する検索項目数

b：FETCH 文で受け取る検索結果中の 1 行のデータ長（実際に取得する各列の長さの合計。単位はバイト）

c：SELECT 句で指定する検索項目で，BINARY 型を指定した選択式の数

## 15.2.11 インメモリデータ処理で必要なメモリ所要量

インメモリデータ処理で必要なメモリ所要量は次に示す計算式で求められます。

HiRDB/ パラレルサーバの場合は，サーバマシンごとにインメモリ化する RD エリアを見積もってください。

**計算式**

**●インメモリデータバッファが使用する共用メモリを実メモリ上に固定しない場合**
計算式 1 ＋ D × 2　（単位：キロバイト）
**●インメモリデータバッファが使用する共用メモリを実メモリ上に固定する場合**
計算式 1 ＋ D ×↑（↑ 2048 ÷ p ↑× p）÷ 1024 ↑　（単位：キロバイト）

**計算式 1**

**●インメモリデータバッファが使用する共用メモリを実メモリ上に固定しない場合**

$$\sum_{i=1}^{n} \uparrow \{704+32\times A+48+448\times B+2048+C\times B\} \div 1024 \uparrow$$ （単位：キロバイト）

**●インメモリデータバッファが使用する共用メモリを実メモリ上に固定する場合**

$$\sum_{i=1}^{n} \uparrow \{\uparrow (704+32\times A+48+448\times B+2048+C\times B) \div p \uparrow \times p\} \div 1024 \uparrow$$ （単位：キロバイト）

n：インメモリ RD エリアの数

A：インメモリ RD エリアを構成する HiRDB ファイル数

B：インメモリ RD エリアの総ページ数

C：インメモリ RD エリアのページサイズ

D：計算式 2 の値

p：Windows の Large Page でのページサイズ

pdntenv コマンドで確認できます。

**計算式 2（インメモリデータバッファが使用する共用メモリセグメント数）**

↑計算式 1 の値÷（SHMMAX オペランドの値× 1024）↑

計算式 2 は 1RD エリア当たりの計算式です。インメモリ RD エリアの数だけ計算してください。

計算式 2 で求めた値は，pd_max_resident_rdarea_shm_no オペランドの見積もりに使います。

# 16 RD エリアの容量の見積もり

この章では，各 RD エリアの容量の見積もり方法について説明します。


16.1 ユーザ用 RD エリアの容量の見積もり

16.2 データディクショナリ用 RD エリアの容量の見積もり

16.3 マスタディレクトリ用 RD エリアの容量の見積もり

16.4 データディレクトリ用 RD エリアの容量の見積もり

16.5 データディクショナリ LOB 用 RD エリアの容量の見積もり

16.6 ユーザ LOB 用 RD エリアの容量の見積もり

16.7 レジストリ用 RD エリアの容量の見積もり

16.8 レジストリ LOB 用 RD エリアの容量の見積もり

16.9 リスト用 RD エリアの容量の見積もり

# 16.1 ユーザ用 RD エリアの容量の見積もり

ここでは，ユーザ用 RD エリアの容量の見積もり方法について説明します。

## 16.1.1 ユーザ用 RD エリアの容量の計算方法

### (1) ユーザ用 RD エリアの容量の求め方

ユーザ用 RD エリアの容量は，次に示す計算式で求めます。

**計算式**

ユーザ用 RD エリアの容量（単位：バイト）
＝ユーザ用 RD エリアのページ長※1 ×ユーザ用 RD エリアの総ページ数※2

注※ 1

データベース初期設定ユティリティ（pdinit）又はデータベース構成変更ユティリティ（pdmod）の create rdarea 文で指定するページ長です。

注※ 2

「(2) ユーザ用 RD エリアの総ページ数を求める計算式」を参照してください。

### (2) ユーザ用 RD エリアの総ページ数を求める計算式

ユーザ用 RD エリアの総ページ数は，次に示す計算式で求めます。

**計算式**

ユーザ用 RD エリアの総ページ数（単位：ページ）
＝ディレクトリページ部分の総ページ数＋データページ部分の総ページ数

#### (a) ディレクトリページ部分の総ページ数の計算式

ディレクトリページ部分の総ページ数（単位：ページ）＝

$$6\times(n+1)+\uparrow 20480\div P\uparrow\times 2+\sum_{i=1}^{n}\{\uparrow d_i\div b\uparrow+\uparrow di\div f\uparrow\}$$

n：ユーザ用 RD エリアを構成する HiRDB ファイル数

P：ユーザ用 RD エリアのページ長（バイト）

b：↓（P − 20）÷（↑ S ÷ 32 ↑× 8 ＋ 56）↓

f：↓（125 × P）÷（16 × b）↓× b

$d_i$：データベース初期設定ユティリティ（pdinit）又はデータベース構成変更ユティリティ（pdmod）の create rdarea 文で指定した各 HiRDB ファイルのセグメント数

S：データベース初期設定ユティリティ（pdinit）又はデータベース構成変更ユティリティ（pdmod）の create rdarea 文で指定した 1 セグメントのページ数（セグメントサイズ）

**(b) データページ部分の総ページ数の計算式**

```
データページ部分の総ページ数（単位：ページ）＝
     e
     Σ {↑αi÷S↑×S}
    i=1
     e
   ＋Σ {↑βi÷S↑×S}
    i=1
     k
   ＋Σ {↑（γi＋1）÷S↑×S}
    i=1
```

e：ユーザ用 RD エリアに格納する表の総数

k：ユーザ用 RD エリアに格納するインデクスの総数

S：データベース初期設定ユティリティ（pdinit）又はデータベース構成変更ユティリティ（pdmod）の create rdarea 文で指定した 1 セグメントのページ数（セグメントサイズ）

$\alpha_i$：各表の分岐するとした BINARY 列以外の列を格納するために必要なページ数
「16.1.2　表の格納ページ数の計算方法」を参照してください。

$\beta_i$：各表の分岐するとした BINARY 列を格納するために必要なページ数
「16.1.2　表の格納ページ数の計算方法」を参照してください。

$\gamma_i$：各インデクスを格納するために必要なページ数
「16.1.3　インデクスの格納ページ数の計算方法」を参照してください。

## 16.1.2　表の格納ページ数の計算方法

CREATE TABLE で FIX 指定をするかどうかによって，表の格納ページ数の計算方法が異なります。それぞれの計算方法を (1) ～ (2) に説明します。(1) 及び (2) の計算式中で使用する変数については，(3) で説明しています。表の格納ページ数の計算例については，(7) で説明しています。また，リバランス機能を使用する場合の RD エリア容量見積もりを (6) で説明しています。

なお，表を横分割する場合，格納 RD エリアごとにページ数を求めてください。

### (1) FIX 指定がない場合

FIX 指定がない場合の表の格納ページ数は，次に示す計算式で求めます。

**計算式**

```
表の格納ページ数＝
      分岐するとした BINARY 列以外を格納するページ数
    ＋分岐するとした BINARY 列を格納するページ数          （単位：ページ）
```

・分岐するとした BINARY 列以外を格納するページ数

$$\left\lceil \frac{(P+SPN1+\sum_{i=1}^{n} PS_i) \times g}{g-\left\lfloor \frac{g \times h}{100} \right\rfloor} \right\rceil$$

・分岐するとした BINARY 列を格納するページ数

SPN2

### (a) P の求め方

P の求め方を次に示します。なお，P の分母の括弧部は 1 ページに格納される行数であり，最小 1，最大 255 とします。

$$P = \left\lceil \frac{a}{\left( MIN\left( 255, \left\lfloor \frac{\left\lfloor \frac{b \times (100-c)}{100} \right\rfloor - 48}{\left\lceil \frac{\sum_{i=1}^{f} d_i}{2} \right\rceil \times 2+8+2 \times f} \right\rfloor \right) \right)} \right\rceil$$

なお，$\left\lfloor \frac{b \times (100-c)}{100} \right\rfloor - 48 \geqq \left\lceil \frac{\sum_{i=1}^{f} d_i}{2} \right\rceil \times 2+8+2 \times f$

を満たすように b，c の値を決定してください。

### (b) $PS_i$ の求め方

$PS_i$ の求め方を次に示します。次に示す計算式で各 $PS_i$ を計算し，その総和を求めてください。なお，n は「表 16-2　可変長文字列型のデータ長一覧（抽象データ型及び繰返し列を除く）」に該当する列の数を示しています。

$PS_i = a \times \uparrow e_i \div (b - 62) \uparrow$

## (2) FIX 指定がある場合

FIX 指定がある場合の表の格納ページ数は，次に示す計算式で求めます。

**計算式**

$$\text{表の格納ページ数} = \left\lceil \frac{Q \times g}{g-\left\lfloor \frac{g \times h}{100} \right\rfloor} \right\rceil$$

（単位：ページ）

### (a) Q の求め方

Q の求め方を次に示します。なお，Q の分母の括弧部は 1 ページに格納される行数であり，最小 1，最大 255 とします。

$$Q = \left\lceil \frac{a}{\left( MIN\left( 255, \left\lfloor \frac{\left\lfloor \frac{b \times (100 - c)}{100} \right\rfloor - 48}{\left\lceil \frac{\sum_{i=1}^{f} d_i}{2} \right\rceil \times 2 + 6} \right\rfloor \right) \right)} \right\rceil$$

なお，$\left\lfloor \frac{b \times (100 - c)}{100} \right\rfloor - 48 \geqq \left\lceil \frac{\sum_{i=1}^{f} d_i}{2} \right\rceil \times 2 + 6$

を満たすようにｂ，ｃの値を決定してください。
ｂの値を決定する場合，マニュアル「HiRDB Version 8 SQLリファレンス」のCREATE TABLEの共通規則に記載されている，列の長さの合計による制限も満たす必要があります。

## (3) 計算式中で使用する変数

a：表に格納する行の総数（件）

b：ユーザ用 RD エリアのページ長（バイト）

c：CREATE TABLE で指定する未使用領域の比率（%）
未使用領域の比率を指定しない場合は，30%を仮定して計算します。

$d_i$：各列のデータ長（バイト）
「表 16-1　データ長一覧」を参照して，すべての列について求めてください。
抽象データ型の列のデータ長については，「(4) 抽象データ型の列のデータ長の求め方」を参照してください。
繰返し列のデータ長については，「(5) 繰返し列のデータ長の求め方」を参照してください。

$e_i$：列のデータ長の平均値（バイト）

- 既定義型で定義された列の場合，「表 16-2　可変長文字列型のデータ長一覧（抽象データ型及び繰返し列を除く）」を参照して，表中に示したデータ型の列についてだけ求めてください。
- 抽象データ型で定義された列の場合，「表 16-3　可変長文字列型のデータ長一覧（抽象データ型の場合）」を参照して，表中に示したデータ型の列についてだけ求めてください。
- 繰返し列の場合，「表 16-4　可変長文字列型のデータ長一覧（繰返し列の場合）」を参照して，表中に示したデータ型の列についてだけ求めてください。

f：表に定義する列の総数（個）

g：表を格納する RD エリアのセグメントサイズ（ページ）

h：CREATE TABLE で指定するセグメント内の空きページ比率（%）
セグメント内の空きページ比率を指定しない場合は，10%を仮定して計算します。ここでいう空きページとは，未使用ページのことです。

SPN1：分岐するとした列（BINARY 以外）を格納するページ数

なお，分岐する条件については「表 16-1　データ長一覧」の注※ 5 で説明しています。

SPN1 ＝
f
Σ ↑分岐するとした $d_i$ の値÷（b － 61）↑× a × SF
i=1

SPN2：分岐するとした BINARY 列を格納するページ数

なお，分岐する条件については「表 16-1　データ長一覧」の注※ 5 で説明しています。

SPN2 ＝ SPN2A ＋ SPN2B ＋ SPN2C

・INSERT SQL による分岐ページ数
SPN2A ＝
f
{Σ ↓分岐するとした $d_i$ の値÷（b － 59）↓× a ＋ A} × SF
i=1

・pdload 又は pdrorg による分岐ページ数
分岐するとした $d_i$ の値＞（b － 2853）÷ 255 のとき
SPN2B ＝
f
↑ {Σ（分岐するとした $d_i$ の値＋ 11）× a} ÷（b － 48）↑× SF
i=1
分岐するとした $d_i$ の値≦（b － 2853）÷ 255 のとき
SPN2C ＝
↑ a ÷ 255 ↑× SF

A の計算式を次に示します。

$$A = \left\uparrow \frac{a}{\mathrm{MIN}\left( 255, \left\downarrow \frac{b-48}{\sum_{i=1}^{f} \{ d_i - (b-59) \times \left\downarrow \frac{\text{分岐するとしたdiの値}}{b-59} \right\downarrow + 11 \}} \right\downarrow \right)} \right\uparrow$$

f：分岐するとした列の数

SF：1.3

ただし，次に示す場合は 1.3 より大きくしてください。

- 抽象データ型の列を大量に更新する場合
- 繰返し列に対して要素のデータ長が大きくなる更新又は要素数が増える更新を大量に実行する場合
- VARCHAR，NVARCHAR，MVARCHAR，又は BINARY 型の列に対してデータ長が大きくなる更新を大量に実行する場合
- BINARY 型の列に，データ長が大きくなる更新を大量に実行する場合
- 列ごとのデータ抑制が実行された列に対して，データ長が大きくなる更新を大量に実行する場合
- 前記以外のデータ型で，NULL 値から非 NULL 値への更新を大量に実行する場合

**表 16-1 データ長一覧**

| 分類 | データ型及び条件 | | | | データ長（単位：バイト） |
|---|---|---|---|---|---|
| 数値データ | INTEGER | | | | 4 |
| | SMALLINT | | | | 2 |
| | LARGE DECIMAL(m,n) ※1 | | | | ↓ m ÷ 2 ↓ ＋ 1 ※2 |
| | FLOAT 又は DOUBLE PRECISION | | | | 8 |
| | SMALLFLT 又は REAL | | | | 4 |
| 文字データ | CHARACTER(n) | | | | n ※3 |
| | VARCHAR(n) | d ≦ 255 | 繰返し列の要素 | | d ＋ 2 |
| | | | 上記以外 | | d ＋ 1 |
| | | d ≧ 256 | | | 6 |
| | VARCHAR(n) ノースプリット オプション 指定あり | n ≦ 255 | 抽象データ型の属性 | | d ＋ 3 |
| | | | 繰返し列の要素 | | d ＋ 2 |
| | | | 上記以外 | | d ＋ 1 |
| | | n ≧ 256 | 分岐する場合※5 | | 6 |
| | | | 分岐しない場合※5 | 抽象データ型の属性 | d ＋ 3 |
| | | | | 繰返し列の要素 | d ＋ 2 |
| | | | | 上記以外 | d ＋ 3 |
| 各国文字データ | NCHAR(n) 又は NATIONAL CHARACTER(n) | | | | 2 × n ※4 |
| | NVARCHAR(n) | d ≦ 127 | 繰返し列の要素 | | 2 × d ＋ 2 |
| | | | 上記以外 | | 2 × d ＋ 1 |
| | | d ≧ 128 | | | 6 |
| | NVARCHAR(n) ノースプリット オプション 指定あり | n ≦ 127 | 抽象データ型の属性 | | 2 × d ＋ 3 |
| | | | 繰返し列の要素 | | 2 × d ＋ 2 |
| | | | 上記以外 | | 2 × d ＋ 1 |
| | | n ≧ 128 | 分岐する場合※5 | | 6 |
| | | | 分岐しない場合※5 | 抽象データ型の属性 | 2 × d ＋ 3 |
| | | | | 繰返し列の要素 | 2 × d ＋ 2 |
| | | | | 上記以外 | 2 × d ＋ 3 |

| 分類 | データ型及び条件 | | | | データ長（単位：バイト） |
|---|---|---|---|---|---|
| 混在文字データ | MCHAR(n) | | | | n※3 |
| | MVARCHAR(n) | d≦255 | 繰返し列の要素 | | d＋2 |
| | | | 上記以外 | | d＋1 |
| | | d≧256 | | | 6 |
| | MVARCHAR(n) ノースプリット オプション 指定あり | n≦255 | 抽象データ型の属性 | | d＋3 |
| | | | 繰返し列の要素 | | d＋2 |
| | | | それ以外 | | d＋1 |
| | | n≧256 | 分岐する場合※5 | | 6 |
| | | | 分岐しない場合※5 | 抽象データ型の属性 | d＋3 |
| | | | | 繰返し列の要素 | d＋2 |
| | | | | 上記以外 | d＋3 |
| 日付データ | DATE | | | | 4 |
| 時刻データ | TIME | | | | 3 |
| 日間隔データ | INTERVAL YEAR TO DAY | | | | 5 |
| 時間隔データ | INTERVAL HOUR TO SECOND | | | | 4 |
| 時刻印データ | TIMESTAMP(n) | | | | 7＋（n÷2） |
| 長大データ | BLOB | | | | 9 |
| バイナリデータ | BINARY(n) | n≦255 | | | d＋3 |
| | | n≧256 | 分岐する場合※5 | | 15 |
| | | | 分岐しない場合※5 | | d＋3 |

d：実際のデータ長（文字数）

m，n：正の整数

注※1

全体のけた数が m けたで，小数点以下のけた数が n けたの固定小数点数です。m を省略した場合は 15 を仮定します。

注※2

表定義時に表オプションに SUPPRESS DECIMAL を指定した場合，データ長は「↓ k ÷ 2 ↓＋2」になります。k は，格納時の有効けた数（先頭の 0 の部分を除いたけた数）を示します。なお，次に示す場合は SUPPRESS DECIMAL を使用しないでください。計算式中の a は，SUPPRESS DECIMAL 又は列データ抑制指定を使用しない場合の，表中の列のデータ長の合計値です。

**32717 ＜（a ＋表中の列数×** 2 ＋ 8**）**

注※3

列データ抑制指定をして，データ抑制された場合，n は「n － b ＋ 4」になります。なお，データ抑制は，列データ抑制指定時，列データの最後の文字が空白の場合，その最後の文字と連続している半角の空白が 4 文字以上あるときだけ実行されます。b は，列データの最後の文字と連続している空白の数を示します。

ただし，列データ抑制指定をして，データ抑制されなかった場合は，列ごとに 1 バイトの付加情報が追加されます。

なお，次に示す場合は列データ抑制指定を使用しないでください。計算式中の a は，SUPPRESS DECIMAL 又は列データ抑制指定を使用しない場合の，表中の列のデータ長の合計値です。

**32717 ＜（a ＋表中の列数×** 2 ＋ 8**）**

注※4

列データ抑制指定をして，データ抑制された場合，2 × n は「2 × n − 2 × b + 5」になります。なお，データ抑制は，列データ抑制指定時，列データの最後の文字が空白の場合，その最後の文字と連続している全角の空白が 3 文字以上あるときだけ実行されます。b は，列データの最後の文字と連続している空白の数を示します。
ただし，列データ抑制指定をして，データ抑制されなかった場合は，列ごとに 1 バイトの付加情報が追加されます。
なお，次に示す場合は列データ抑制指定を使用しないでください。計算式中の a は，SUPPRESS DECIMAL 又は列データ抑制指定を使用しない場合の，表中の列のデータ長の合計値です。
**32717 ＜（a ＋表中の列数×** 2 ＋ 8**）**

注※ 5
通常は分岐しない場合で計算します。次に示す計算式が成立する場合に分岐します。

$$BL > \text{ページ長} - 50$$
$$BL(\text{バイト}) = \sum_{i=1}^{f} di + 2 \times f + 6$$

この分岐条件が成立した場合は，分岐しないとした列を列番号の小さい方から分岐条件が成立しなくなるまで分岐するとして BL を計算し直してください。

**表 16-2　可変長文字列型のデータ長一覧（抽象データ型及び繰返し列を除く）**

| データ型 | | データ長（バイト） |
|---|---|---|
| VARCHAR(n) | d ≧ 256 | d ＋ 2 |
| | ノースプリットオプション指定あり | 0 |
| NVARCHAR(n) | d ≧ 128 | 2 × d ＋ 2 |
| | ノースプリットオプション指定あり | 0 |
| MVARCHAR(n) | d ≧ 256 | d ＋ 2 |
| | ノースプリットオプション指定あり | 0 |

d：実際のデータ長（文字数）

## （4）抽象データ型の列のデータ長の求め方

抽象データ型の列のデータ長 $d_i$ は，次に示す計算式で求めます。

**計算式**

$$di = \sum_{k=1}^{h} ADT_k + 5$$

h：抽象データ型の継承の数（個）
継承なしの場合は 1
CREATE TYPE 文で UNDER オペランドを指定して別の抽象データ型を継承した場合，最も上位の抽象データ型を h 番目，最も下位の抽象データ型を 1 番目としてください。

$ADT_k$：抽象データ型のデータ長（バイト）
次に示す計算式で求めてください。

$$ADT_k = \sum_{i=1}^{m} att_j + 10 + 2 \times m$$

m：抽象データ型の全属性数（個）

$att_j$：抽象データ型の各属性のデータ長（バイト）
継承がない場合は，m=1 であり，$ADT_1$ を計算します。

各属性のデータ長については，「表 16-1　データ長一覧」を参照してください。ただし，データ型が「表 16-3　可変長文字列型のデータ長一覧（抽象データ型の場合）」で示す条件を満たしている場合は，表 16-3 に従ってデータ長を計算してください。

また，対応する $atte_j$ の値を次に示す計算式に代入して，分岐行格納ページ数 ADTLS を P に加算してください。

$$ADTLS = \sum_{i=1}^{h} \uparrow atte_j \div (b-62) \uparrow \times a$$

属性が抽象データ型で定義されている場合は，次に示す計算式で属性のデータ長を求めてください。

$$att_j（バイト） = \sum_{k=1}^{h} ADT_k + 5$$

**表 16-3　可変長文字列型のデータ長一覧（抽象データ型の場合）**

| データ型 | 条　件 | データ長 $att_j$（バイト） | 分岐部分のデータ長 $atte_j$（バイト） |
|---|---|---|---|
| VARCHAR(n) | d ≧ 256 | 8 | d ＋ 2 |
| | ノースプリットオプション指定あり | d ＋ 3 | 0 |
| NVARCHAR(n) | d ≧ 128 | 8 | 2 × d ＋ 2 |
| | ノースプリットオプション指定あり | 2 × d ＋ 3 | 0 |
| MVARCHAR(n) | d ≧ 256 | 8 | d ＋ 2 |
| | ノースプリットオプション指定あり | d ＋ 3 | 0 |

d：実際のデータ長（文字数）

## (5) 繰返し列のデータ長の求め方

繰返し列のデータ長は，次に示す計算式で求めます。

**計算式**

$$d_i = 4 + (el_i + 1) \times en_i$$

$el_i$：繰返し列のデータ長
「表 16-1　データ長一覧」から求めてください。
ただし，可変長文字列型の場合は，「表 16-4　可変長文字列型のデータ長一覧（繰返し列の場合）」から求めてください。

$en_i$：繰返し列の平均要素数

**表 16-4　可変長文字列型のデータ長一覧（繰返し列の場合）**

| データ型 | 条件 | データ長 $el_i$（バイト） | 分岐部分のデータ長 $es_j$（バイト） |
|---|---|---|---|
| VARCHAR(n) | d ≧ 256 | 5 | d ＋ 2 |
| | ノースプリットオプション指定あり | d ＋ 2 | 0 |
| NVARCHAR(n) | d ≧ 128 | 5 | 2 × d ＋ 2 |
| | ノースプリットオプション指定あり | 2 × d ＋ 2 | 0 |
| MVARCHAR(n) | d ≧ 256 | 5 | d ＋ 2 |
| | ノースプリットオプション指定あり | d ＋ 2 | 0 |

d：実際のデータ長（文字数）

可変長文字列型の繰返し列で，$el_i$ の値が表 16-4 の条件を満たす列について，次に示す計算式の値を P に加算してください。

$$\uparrow \sum_{i=1}^{m} \{ es_i \times en_i + 14 \times (en_i - 1) \} \div (b - 62) \uparrow \times a$$

m：表 16-4 の条件を満たす可変長文字列型の繰返し列数

$es_i$：1 要素当たりの実際のデータ長の平均値

「表 16-2　可変長文字列型のデータ長一覧（抽象データ型及び繰返し列を除く）」に示したデータ長を適用します。

## (6) リバランス機能を使用する場合のエリア容量見積もり

ハッシュ関数 HASHA，HASHB，HASHC，HASHD，HASHE，HASHF を使用した分割表の場合，データは 1,024 個のハッシュ要素値に分けられ，値ごとに別々のセグメントに格納されます。

各分割 RD エリアには，平均（1024 ÷分割数）のハッシュ要素数のデータが格納されます。このため，各 RD エリアには，少なくともその RD エリアに格納される要素数分のセグメントを割り当てる必要があります。

リバランス機能を使用する場合の RD エリア容量は次のように見積もります。

1. データ件数 N，行長 L，ページ長 P から，必要な総セグメント数 Sn を見積もります。
2. RD エリア当たりに必要なセグメント数 Ssn を見積もります。

   Ssn ＝↑ Sn ÷ Srn ↑× Srn
   Srn：↑ 1024 ÷ Dvn ↑
   Dvn：RD エリア分割数
3. 余裕値を考慮して，RD エリア当たりの使用中セグメント数 S を見積もります。

   S ＝↑（Ssn × K）÷ Srn ↑× Srn
   K：係数（例：余裕率 20% の場合，1.2）

## (7) 表の格納ページ数の計算例

### (a) 例題

次に示す在庫表の表格納ページ数を求めます。

| 品番 | 商品名 | 規格 | 単価 | 数量 | 原価 |
|---|---|---|---|---|---|
| 20180 | 掃除機 | C20 | 20000 | 26 | 15000 |
| 20190 | 掃除機 | C77 | 28000 | 105 | 23000 |
| 20130 | 冷蔵庫 | P10 | 30000 | 70 | 25000 |
| 20220 | テレビ | K18 | 35000 | 12 | 30000 |
| 20200 | 掃除機 | C89 | 35000 | 30 | 30000 |
| 20140 | 冷蔵庫 | P23 | 35000 | 60 | 30000 |
| 20280 | アンプ | L10 | 38000 | 200 | 33000 |
| 20150 | 冷蔵庫 | P32 | 48000 | 50 | 43000 |
| 20290 | アンプ | L50 | 49800 | 260 | 45000 |
| 20230 | テレビ | K20 | 50000 | 15 | 45000 |
| 20160 | 冷蔵庫 | P35 | 55800 | 120 | 50000 |

**計算条件**

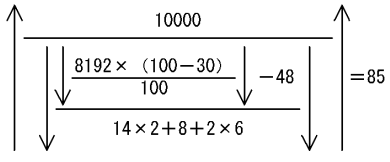
1. 表に格納する行の総数：10000 件
2. ユーザ用 RD エリアのページ長：8192 バイト
3. CREATE TABLE で指定する未使用領域の比率：30%
4. 列数：6 列
5. 表を格納する RD エリアのセグメントサイズ：100 ページ
6. CREATE TABLE で指定するセグメント内の空きページ比率：40%
7. 列のデータ型：次に示します。
   品番：CHARACTER(5)
   商品名：NCHAR(4)
   規格：CHARACTER(3)
   単価：INTEGER
   数量：INTEGER
   原価：INTEGER

**FIX 指定がない場合**

1. **行長の計算**

5（品番）＋ 2 × 4（商品名）＋ 3（規格）＋ 4（単価）＋ 4（数量）＋ 4（原価）＝ 28 バイト

2. **P の計算**



3. **表の格納ページ数の計算**

$$\left\lceil \frac{85 \times 100}{100 - \left\lfloor \frac{100 \times 40}{100} \right\rfloor} \right\rceil = 142\text{ページ}$$

FIX **指定がある場合**

1. **行長の計算**

5（品番）＋ 2 × 4（商品名）＋ 3（規格）＋ 4（単価）＋ 4（数量）＋ 4（原価）＝ 28 バイト

2. Q **の計算**

$$\left\lceil \frac{10000}{\left\lfloor \frac{8192 \times (100-30)}{100} \right\rfloor - 48 - (14 \times 2 + 6)} \right\rceil = 60$$

3. **表の格納ページ数の計算**

$$\left\lceil \frac{60 \times 100}{100 - \left\lfloor \frac{100 \times 40}{100} \right\rfloor} \right\rceil = 100\text{ページ}$$

## 16.1.3 インデクスの格納ページ数の計算方法

インデクスの格納ページ数の計算方法を (1) で説明します。(1) の計算式中で使用する変数については (2) で説明しています。インデクスの格納ページ数の計算例については (3) で説明しています。

なお，CREATE TABLE でクラスタキーを指定する場合，インデクスの格納ページ数を求める方法と同じ方法で，クラスタキーの格納ページ数を求めてください。

また，インデクスを横分割する場合，格納 RD エリアごとにページ数を求めてください。

**注意事項**

インデクスページスプリットが発生すると，インデクスページ内のキーの格納比率を 50：50 にして二つのインデクスページに分割します。このため，インデクスの追加又は更新が多く発生すると，インデクスの格納ページ数は最大で見積もり式の 2 倍の容量が必要となります。また，最大キーが格納されたリーフページのインデクスページスプリットは，UAP からの INSERT であっても PCTFREE オペランドの値が考慮されます。

なお，インデクスページスプリットの発生回数を削減する方法の一つにアンバランスインデクススプリットがあります。インデクスページスプリット及びアンバランスインデクススプリットについては，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

### (1) 計算方法

インデクスの格納ページ数は，次に示す計算式で求めます。

**計算式**

$$\text{インデクスの格納ページ数（単位：ページ）} = \sum_{i=1}^{n} Pi + P_d$$

Pi は，**計算式 1** に示す漸化式から求めます。

$P_n = 1$ となるまで $P_{i+1}$ の計算をし，その計算結果の総計を求めてください。

$P_d$ は，キー値の重複が 201 以上の場合に計算します。$P_d$ の求め方を**計算式 2** に示します。

**計算式 1**

$$P_1 = \left\lceil \frac{c-h}{\mathrm{MAX}\left(1, \left\lfloor \frac{\left\lfloor \frac{a\times(100-b)}{100} \right\rfloor - g - 68}{18+g+4\times d} \right\rfloor\right)} \right\rceil + \left\lceil \frac{e}{\mathrm{MAX}\left(1, \left\lfloor \frac{\left\lfloor \frac{a\times(100-b)}{100} \right\rfloor - g - 68}{18+g} \right\rfloor\right)} \right\rceil + \left\lceil \frac{h}{\mathrm{MAX}\left(1, \left\lfloor \frac{\left\lfloor \frac{a\times(100-b)}{100} \right\rfloor - g - 68}{14+g} \right\rfloor\right)} \right\rceil$$

$$P_{i+1} = \left\lceil \frac{P_i}{\mathrm{MAX}\left(1, \left\lfloor \frac{\left\lfloor \frac{a\times(100-b)}{100} \right\rfloor - g - 68}{14+g} \right\rfloor + 1\right)} \right\rceil$$

**計算式 2**

$$Pd = \left\{ \left\lceil \frac{f}{\left\lfloor \frac{\left\lfloor \frac{a\times 95}{100} \right\rfloor - 70}{4} \right\rfloor} \right\rceil + 1 \right\} \times e$$

**繰返し列を含むインデクスの場合の 1 行当たりの繰返し要素の重複数について**

繰返し列を含むインデクスの場合，1 行当たりの繰返し要素の重複数は，次に示す計算式の値を超えないようにしてください。

重複数＝↓（↓ a × 0.95 ↓－ 82）÷ 4 ↓－ 1

## （2）計算式中で使用する変数

a：ユーザ用 RD エリアのページ長（バイト）

b：CREATE INDEX で指定する未使用領域の比率※1（%）

c：キー値の重複が 200 以下のキーの種類の個数（個）※2，3，4

d：キー値の重複が 200 以下のキーの重複数の平均値（個）※3，5

e：キー値の重複が 201 以上のキーの種類の個数（個）※3，4

f：キー値の重複が 201 以上のキーの重複数の平均値（個）※3，5

g：DB 格納キー長※6（バイト）

h：次に示すどちらかを代入します。

- ユニークインデクスの場合：ナル値以外のキーの種類の個数（個）
  なお，複数列インデクスの場合は，構成列にナル値を含まない全キー数となります。
- ユニークインデクス以外の場合：0

注※ 1

未使用領域の比率を指定しない場合は，30%を仮定して計算します。また，クラスタキーを指定する場合は，CREATE TABLE で指定する未使用領域の比率とします。

注※ 2

ユニークインデクスの重複がないキーを含める必要があります。

注※ 3

c × d ＋ e × f の値が，インデクスのキーの総数以上になるように計算してください。

注※ 4

ユニークインデクスの重複があるキー（キー値にナル値を含むことで重複するキー）を含める必要があります。

注※ 5

小数点未満は整数値に切り上げてください。

注※ 6

「表 16-5　インデクスのキー長一覧」を参照してください。DB 格納キー長は，次に示す計算式で求めます。

- 単一列インデクス及び固定長複数列インデクスの場合
  **↑キー長÷ 4 ↑× 4**
- 可変長複数列インデクスで，かつキー長 255 バイト以下の場合
  **↑（キー長＋ 1）÷ 4 ↑× 4**

- 可変長複数列インデクスで，かつキー長 256 バイト以上の場合
  **↑（キー長＋ 2）÷ 4 ↑× 4**

ただし，複数列インデクスのキー長は，「表 16-5　インデクスのキー長一覧」を基に全構成列のキー長を加算したものとなります。

**表 16-5　インデクスのキー長一覧**

| 分類 | データ型 | キー長（単位：バイト） | | | | | |
|---|---|---|---|---|---|---|---|
| | | キー長が 255 バイト以下 | | | キー長が 256 バイト以上 | | |
| | | 単一列インデクス | 複数列インデクス | | 単一列インデクス | 複数列インデクス | |
| | | | 固定長※1 | 可変長※2 | | 固定長※1 | 可変長※2 |
| 数値データ | INTEGER | 4 | 5 | 6 | － | 5 | 7 |
| | SMALLINT | 2 | 3 | 4 | － | 3 | 5 |
| | LARGE DECIMAL (m,n)※3 | ↓ m ÷ 2 ↓ ＋ 1 | ↓ m ÷ 2 ↓＋ 2 | ↓ m ÷ 2 ↓＋ 3 | － | ↓ m ÷ 2 ↓ ＋ 2 | ↓ m ÷ 2 ↓ ＋ 4 |
| | FLOAT 又は DOUBLE PRECISION | 8 | × | × | － | × | × |
| | SMALLFLT 又は REAL | 4 | × | × | － | × | × |
| 文字データ | CHARACTER (n) | n | n ＋ 1 | n ＋ 2 | n | n ＋ 1 | n ＋ 3 |
| | VARCHAR(n) | a ＋ 1 | － | a ＋ 2 | a ＋ 2 | － | a ＋ 3 |
| 各国文字データ | NCHAR(n) 又は NATIONAL CHARACTER (n) | 2 × n | 2 × n ＋ 1 | 2 × n ＋ 2 | 2 × n | 2 × n ＋ 1 | 2 × n ＋ 3 |
| | NVARCHAR (n) | 2 × b ＋ 1 | － | 2 × b ＋ 2 | 2 × b ＋ 2 | － | 2 × b ＋ 3 |
| 混在文字データ | MCHAR(n) | n | n ＋ 1 | n ＋ 2 | n | n ＋ 1 | n ＋ 3 |
| | MVARCHAR (n) | a ＋ 1 | － | a ＋ 2 | a ＋ 2 | － | a ＋ 3 |
| 日付データ | DATE | 4 | 5 | 6 | － | 5 | 7 |
| 時刻データ | TIME | 3 | 4 | 5 | － | 4 | 6 |
| 日間隔データ | INTERVAL YEAR TO DAY | 5 | 6 | 7 | － | 6 | 8 |
| 時間隔データ | INTERVAL HOUR TO SECOND | 4 | 5 | 6 | － | 5 | 7 |
| 時刻印データ | TIMESTAMP(p) | 7 ＋（p ÷ 2） | 8 ＋（p ÷ 2） | 9 ＋（p ÷ 2） | － | 8 ＋（p ÷ 2） | 10 ＋（p ÷ 2） |

（凡例）
a：実際のデータ長

b：実際の文字数
m，n，p：正の整数
×：インデクス定義時にエラーになります。
－：該当しません。

注
最初は「キー長が 255 バイト以下」で計算してください。その結果，キー長が 256 バイト以上になる場合は，「キー長が 256 バイト以上」で再計算してください。

注※ 1
構成列が固定長の列だけのインデクスのキー長です。

注※ 2
構成列に可変長の列を含むインデクスのキー長です。

注※ 3
全体のけた数が m けたで，小数点以下のけた数が n けたの固定小数点数です。m を省略した場合は 15 を仮定します。

## (3) インデクスの格納ページ数の計算例

### (a) 例題 1

次に示す在庫表の「品番」をユニークインデクス（重複するキーがない）とする場合のインデクス格納ページ数を求めます。

| 品番 | 商品名 | 規格 | 単価 | 数量 | 原価 |
|---|---|---|---|---|---|
| 20180 | 掃除機 | C20 | 20000 | 26 | 15000 |
| 20190 | 掃除機 | C77 | 28000 | 105 | 23000 |
| 20130 | 冷蔵庫 | P10 | 30000 | 70 | 25000 |
| 20220 | テレビ | K18 | 35000 | 12 | 30000 |
| 20200 | 掃除機 | C89 | 35000 | 30 | 30000 |
| 20140 | 冷蔵庫 | P23 | 35000 | 60 | 30000 |
| 20280 | アンプ | L10 | 38000 | 200 | 33000 |
| 20150 | 冷蔵庫 | P32 | 48000 | 50 | 43000 |
| 20290 | アンプ | L50 | 49800 | 260 | 45000 |
| 20230 | テレビ | K20 | 50000 | 15 | 45000 |
| 20160 | 冷蔵庫 | P35 | 55800 | 120 | 50000 |

**計算条件**

1. インデクスのキーの総数：10,000 件
2. ユーザ用 RD エリアのページ長：8,192 バイト
3. CREATE INDEX で指定する未使用領域の比率：30%
4. インデクスのデータ型：CHARACTER
5. インデクスのキー長：5 バイト
6. キーの重複数：1

**計算式**

ＤＢ格納キー長（ｇ）＝⌈５÷４⌉×４＝８

$$P_1 = \left\lceil \frac{10000-10000}{\mathrm{MAX}\left(1,\ \left\lfloor \frac{\left\lfloor \frac{8192\times(100-30)}{100} \right\rfloor - 8 - 68}{18+8+4\times1} \right\rfloor\right)} \right\rceil + \left\lceil \frac{0}{\mathrm{MAX}\left(1,\ \left\lfloor \frac{\left\lfloor \frac{8192\times(100-30)}{100} \right\rfloor - 8 - 68}{18+8} \right\rfloor\right)} \right\rceil + \left\lceil \frac{10000}{\mathrm{MAX}\left(1,\ \left\lfloor \frac{\left\lfloor \frac{8192\times(100-30)}{100} \right\rfloor - 8 - 68}{14+8} \right\rfloor\right)} \right\rceil$$

＝０＋０＋３９＝３９

$$P_2 = \left\lceil \frac{39}{\mathrm{MAX}\left(1,\ \left\lfloor \frac{\left\lfloor \frac{8192\times(100-30)}{100} \right\rfloor - 8 - 68}{14+8} \right\rfloor + 1\right)} \right\rceil = 1$$

Ｐｄ＝０（キー値の重複が２００以下となるため）

インデクスの格納ページ数＝３９＋１＋０＝　４０ページ

### (b) 例題 2

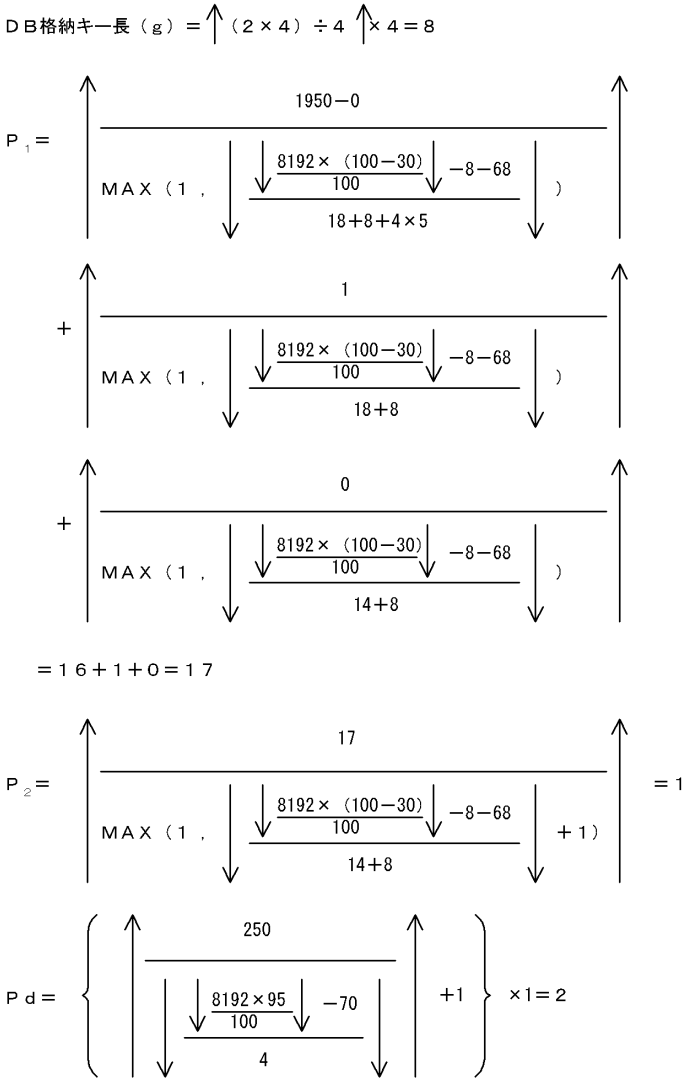
例題 1 に示す在庫表の「商品名」をインデクス（重複するキーがある）とする場合のインデクス格納ページ数を求めます。

**計算条件**

1. インデクスのキーの総数：10,000 件
2. ユーザ用 RD エリアのページ長：8,192 バイト
3. CREATE INDEX で指定する未使用領域の比率：30%
4. インデクスのデータ型：NCHAR
5. インデクスのキー長：4 文字（漢字）
6. キー値の重複が 201 以上のキーの種類の個数
   （このときの平均重複数は 250）：1
7. キー値の重複が 200 以下のキーの種類の個数

（このときの平均重複数は 5）：（10000 − 250）÷ 5 = 1950

**計算式**




インデクスの格納ページ数＝１７＋１＋２＝　２０ページ

次に示す会員表の「性別」と「入会年度」を複数列インデクスとする場合のインデクス格納ページ数を求めます。

| 会員番号 | 氏名 | 年齢 | 性別 | 入会年度 |
|---|---|---|---|---|
| 0001 | 安東洋子 | 18 | F | 1983 |
| 0002 | 浅井順一 | 25 | M | 1967 |
| 0003 | 石井公子 | 24 | F | 1987 |

| 会員番号 | 氏名 | 年齢 | 性別 | 入会年度 |
|---|---|---|---|---|
| 0004 | 石井一郎 | 25 | M | 1964 |
| ・ | ・ | ・ | ・ | ・ |
| ・ | ・ | ・ | ・ | ・ |
| ・ | ・ | ・ | ・ | ・ |
| 1000 | 渡辺英雄 | 30 | M | 1995 |

**計算条件**

1. インデクスのキーの総数：10,000 件
2. ユーザ用 RD エリアのページ長：8,192 バイト
3. CREATE INDEX で指定する未使用領域の比率：30%
4. 1964 年度の入会人数：1,000 人
5. ほかの年度の入会人数：200 人以下
6. 入会年度：1965 ～ 1995 の 31 年間
7. 入会する男女の数は，毎年同数とします。
8. 列のデータ型：次に示します。
   会員番号：CHARACTER(5)
   氏名：NCHAR(4)
   年齢：INTEGER
   性別：CHARACTER(4)
   入会年度：INTEGER

**計算式**

1. 1965 年度以降の 31 年間に入会した人（男，女含めて 200 人以下）のキーの種類の個数（c）は，
   c ＝ 31 × 2 ＝ 62 となります。
2. 重複数の平均値（d）は，d ＝（10000 － 1000）÷ 62 ＝ 146 となります。
3. 1964 年の 1 年間に入会した人（男，女含めて 1000 人）のキーの種類の個数（e）は，e ＝ 2 となります。
4. 重複数の平均値（f）は，f ＝ 1000 ÷ 2 ＝ 500 となります。
5. 性別と入会年度の DB 格納キー長（g）は，
   g ＝↑（4 ＋ 1 ＋ 5）÷ 4 ↑× 4 ＝ 12 となります。

$$P_1 = \left\lceil \frac{62-0}{\mathrm{MAX}\left(1,\ \left\lfloor \frac{\left\lfloor \frac{8192\times(100-30)}{100} \right\rfloor - 12 - 68}{18+12+4\times146} \right\rfloor\right)} \right\rceil + \left\lceil \frac{2}{\mathrm{MAX}\left(1,\ \left\lfloor \frac{\left\lfloor \frac{8192\times(100-30)}{100} \right\rfloor - 12 - 68}{18+12} \right\rfloor\right)} \right\rceil + \left\lceil \frac{0}{\mathrm{MAX}\left(1,\ \left\lfloor \frac{\left\lfloor \frac{8192\times(100-30)}{100} \right\rfloor - 12 - 68}{14+12} \right\rfloor\right)} \right\rceil$$

$= 7 + 1 + 0 = 8$

$$P_2 = \left\lceil \frac{8}{\mathrm{MAX}\left(1,\ \left\lfloor \frac{\left\lfloor \frac{8192\times(100-30)}{100} \right\rfloor - 12 - 68}{14+12} \right\rfloor + 1\right)} \right\rceil = 1$$

$$Pd = \left\{ \left\lceil \frac{500}{\left\lfloor \frac{\left\lfloor \frac{8192\times95}{100} \right\rfloor - 70}{4} \right\rfloor} \right\rceil + 1 \right\} \times 2 = 4$$

インデクスの格納ページ数＝８＋１＋４＝　１３ページ

# 16.2 データディクショナリ用 RD エリアの容量の見積もり

データディクショナリ用 RD エリアは，データベース構成変更ユティリティ（pdmod）の create rdarea 文の指定によって次に示す 2 種類作成できます。

- 通常のデータディクショナリ用 RD エリア
  create rdarea 文に datadictionary，又は datadictionary of routines を指定
- 解析情報表及び運用履歴表を格納するデータディクショナリ用 RD エリア
  create rdarea 文に datadictionary of dbmanagement を指定

上記の RD エリアは，それぞれの種類ごとに容量を見積もる必要があります。

## 16.2.1 通常のデータディクショナリ用 RD エリアの容量の見積もり

### (1) データディクショナリ用 RD エリアの容量の求め方

create rdarea 文に datadictionary 又は datadictionary of routines を指定する場合のデータディクショナリ用 RD エリアの容量は，次に示す計算式で求めます。

**計算式**

データディクショナリ用 RD エリアの容量＝ a × b × 1.3 ＋ c × 125 ＋ 1600000（単位：バイト）

a：データディクショナリ用 RD エリアのページ長※ 1

b：データディクショナリ用 RD エリアの総ページ数※ 2

c：データディクショナリ用 RD エリアのセグメントサイズ※ 3

注※ 1
データベース初期設定ユティリティ（pdinit）又はデータベース構成変更ユティリティ（pdmod）の create rdarea 文で指定するページ長です。

注※ 2
「表の格納ページ数＋インデクスの格納ページ数」です。「(2) 表の格納ページ数の計算方法」及び「(3) インデクスの格納ページ数の計算方法」を参照してください。

注※ 3
データベース初期設定ユティリティ（pdinit）又はデータベース構成変更ユティリティ（pdmod）の create rdarea 文で指定するセグメントサイズです。

### (2) 表の格納ページ数の計算方法

表の格納ページ数（単位：ページ）は，計算式 1 ～計算式 25 の総和になります。

**(a) 計算式 1**

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_TABLES | $\left\uparrow \frac{a}{\left\downarrow \frac{b-48}{\uparrow (h\times 2+i\times 2+e+d+f+u+w+622)\div 2\uparrow \times 2}\right.}\right.$ |
| SQL_COLUMNS | ●DEFAULT句を指定しない場合，及びDEFAULT句に指定した既定値のデータ長が255バイト以下の場合<br>$\left\uparrow \frac{a\times g}{\left\downarrow \frac{b-48}{\uparrow (h\times 2+i+c+j+k+m+n+p+t+y+A+256)\div 2\uparrow \times 2}\right.}\right.$<br>●DEFAULT句に指定した既定値のデータ長が256バイト以上の場合<br>$\left\uparrow \frac{a\times g}{\left\downarrow \frac{b-48}{\uparrow (h\times 2+i+c+j+k+m+n+p+t+245+\downarrow (y+A)\div 32000\downarrow \times 6+8)\div 2\uparrow \times 2}\right.}\right.$<br>$+a\times g\times \uparrow (w+y)\div (b-61)\uparrow$ |
| SQL_DIV_TABLE | $\left\uparrow \frac{a\times C}{\left\downarrow \frac{b-48}{\uparrow (h+i+d+B\times 2+78)\div 2\uparrow \times 2}\right.}\right.$ |

a：表の総数（個）

b：データディクショナリ用 RD エリアのページ長（バイト）

c：列の名称長の平均値（バイト）

d：表を格納する RD エリアの名称長の平均値（バイト）

e：表の注釈長の平均値（バイト）

f：表の横分割条件を指定する列の名称長の平均値（バイト）

g：表の列数の平均値（個）

h：認可識別子の長さの平均値（バイト）

i：表識別子の長さの平均値（バイト）

ｊ：列の注釈長の平均値（バイト）

ｋ：ビュー表の基になる表の認可識別子の長さの平均値（バイト）

ｍ：ビュー表の基になる表の表識別子の長さの平均値（バイト）

ｎ：ビュー表の基になる表の列の名称長の平均値（バイト）

ｐ：ユーザ定義型の名称長の平均値（バイト）

ｑ：表の横分割条件数の平均値（個）

ｔ：PLUGIN 句指定長の平均値（バイト）

ｕ：外部サーバの名称の長さの平均値（バイト）

ｗ：挿入履歴保持列の名称の平均値（バイト）

ｙ：DEFAULT 句に指定した既定値の実長の平均値（バイト）
実長の算出方法は，マニュアル「HiRDB Version 8 UAP 開発ガイド」の「SQL 記述領域に設定するデータコードとデータの長さ」を参照してください。

Ａ：DEFAULT 句に指定した既定値の長さの平均値（バイト）
指定した既定値が定数の場合は，定数表現の長さです。既定値の長さを長く変更する可能性がある場合は変更後の長さを考慮して算出してください。文字型の定数の場合，各国文字列定数を表す N，混在文字列定数を表す M，16 進文字列定数を表す X，引用符（ ’）を長さに含みます。それ以外は指定した既定値のバイト数です。

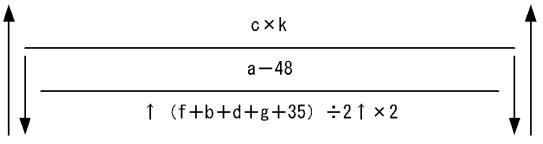
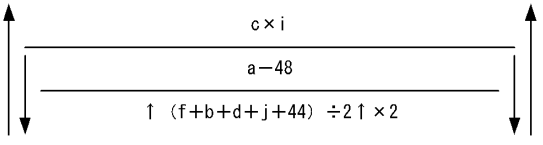
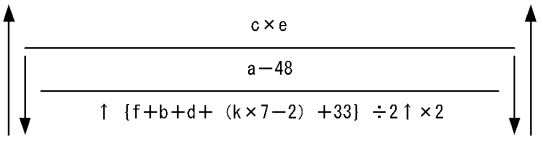
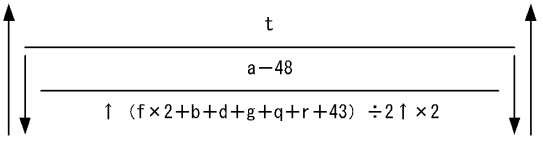
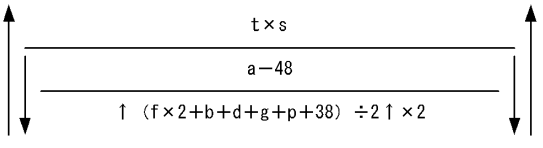
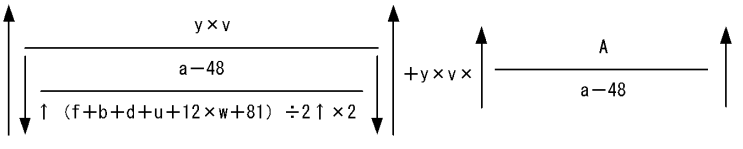
（例）
'HiRDB'：7 バイト
N'Ｈｉ ＲＤＢ'：13 バイト
X'4869524442'：13 バイト
CURRENT_TIME：12 バイト
100：3 バイト

Ｂ：表の横分割条件長の平均値（バイト）

Ｃ：表格納用 RD エリア指定数（個）

**（b）計算式 2**

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_INDEXES | $\left\lceil \frac{c}{\left\lceil \frac{a-48}{\uparrow (f\times 2+b+d+2\times k+j+m+n+146) \div 2 \uparrow \times 2} \right\rceil} \right\rceil$ |

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_INDEX_<br>COLINF |  |
| SQL_DIV_INDEX |  |
| SQL_EXCEPT |  |
| SQL_INDEX_<br>DATATYPE |  |
| SQL_INDEX_<br>FUNCTION |  |
| SQL_INDEX_<br>XMLINF |  |

a：データディクショナリ用 RD エリアのページ長（バイト）

b：表識別子の長さの平均値（バイト）

c：インデクスの総数（個）

d：インデクス識別子の長さの平均値（バイト）

e：1 インデクス当たりのインデクス除外キー値数の平均値（個）

f：認可識別子の長さの平均値（バイト）

g：列の名称長の平均値（バイト）

i：表の横分割条件数の平均値（個）

j：インデクスを格納する RD エリアの名称の長さの平均値（バイト）

k：インデクスを構成する列数の平均値（個）

m：インデクス型名称の長さの平均値（バイト）

n：PLUGIN 句指定長の平均値（バイト）

p：プラグインインデクス適用関数名称の長さの平均値（バイト）

q：抽象データ型名称の長さの平均値（バイト）

r：属性名称の長さの平均値（バイト）

s：1 プラグインインデクス当たりの適用関数の数（個）

t：プラグインインデクスの総数（個）

u：部分構造パス長の平均値（バイト）

v：部分構造インデクスを構成する部分構造パス数の平均値（個）

w：データ長が 256 バイト以上で，かつ分岐するバイナリデータ（部分構造パス用解析ツリー）数の平均値（個）

バイナリデータの格納ページ数の分岐条件については，「表 16-1　データ長一覧」を参照してください。

y：部分構造インデクスの総数（個）

A：部分構造パス用解析ツリー長（バイト）

次の計算式で求められる値になります。

S × 120 ＋ P ＋ L ＋ S × 4 ＋ 32

L：ステップ式の修飾名に指定した局所名の文字列表現長の合計値（バイト）※

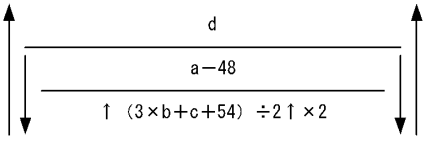
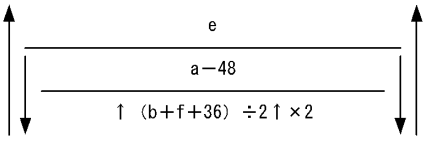
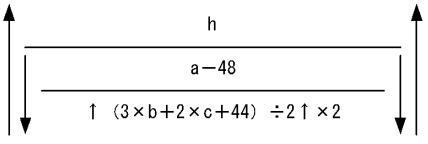
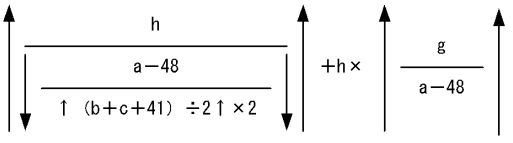
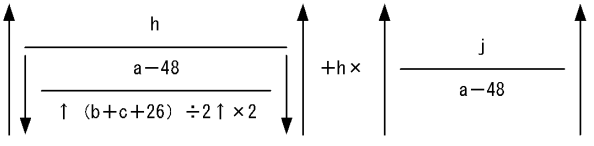
P：接頭辞に関連づけた XML 名前空間 URI の文字列表現長の合計値（バイト）※

接頭辞を省略した場合は，省略時に関連づけられる XML 名前空間 URI

S：ステップ式の指定数（個）

注※　4 の倍数に切り上げてください。

### (c) 計算式 3

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_TABLE_<br>PRIVILEGES |  |
| SQL_RDAREA_<br>PRIVILEGES |  |
| SQL_VIEW_<br>TABLE_USAGE |  |
| SQL_VIEWS |  |
| SQL_VIEW_DEF ※ |  |

a：データディクショナリ用 RD エリアのページ長（バイト）

b：認可識別子の長さの平均値（バイト）

c：表識別子の長さの平均値（バイト）

d：アクセス権限の定義数（個）

- 一つの表に対して n 人に権限を与えた場合は，権限を与えた表数× n 個と計算します。

- PUBLIC に対して権限を与えた場合は，1 人として計算します。
- グループに対して権限を与えた場合は，一つのグループ ID を 1 人として計算します。

e：RD エリアの総数（個）

f：表を格納する RD エリアの名称の長さの平均値（バイト）

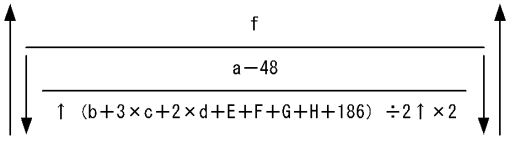
g：ビュー表を定義するときの SQL 文の長さの平均値（バイト）

h：ビュー定義の総数（個）

j：ビュー解析情報の平均長（バイト）
1 ビュー表当たりのビュー解析情報長については，マニュアル「HiRDB Version 8 システム定義」の「ビュー解析情報用バッファ長（pd_view_def_cache_size）の見積もり式」を参照してください。

注※　システムが使用する表です。

**(d) 計算式 4**

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_REFERENTIAL_CONSTRAINTS |  |



E：{e × h ＋ 2 × h ＋（h − 1)} ＋ 1

F：{e × i ＋ 2 × i ＋（i − 1)} ＋ 1

G：{2 × h ＋（h − 1 )} ＋ 1

H：{2 × i ＋（i − 1)} ＋ 1

a：データディクショナリ用 RD エリアのページ長（バイト）

b：制約名称の長さの平均値（バイト）

c：認可識別子の長さの平均値（バイト）

d：表識別子の長さの平均値（バイト）

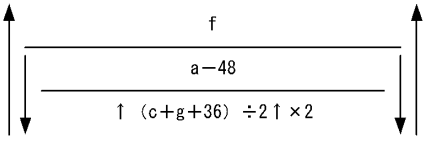
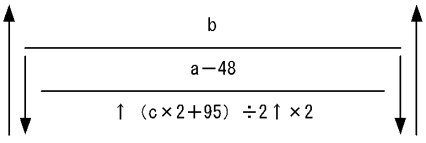
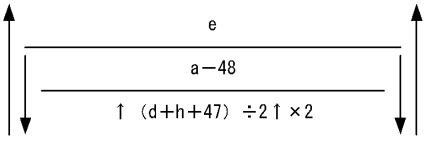
e：外部キーを定義した列名の長さの平均値（バイト）

f：主キーを定義した列名の長さの平均値（バイト）

h：外部キーを構成する列数の平均値（個）

i：主キーを構成する列数の平均値（個）

**（e）計算式 5**

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_PHYSICAL_FILES |  |
| SQL_RDAREAS |  |
| SQL_USERS |  |







a：データディクショナリ用 RD エリアのページ長（バイト）

b：RD エリアの総数（個）

c：RD エリアの名称の長さの平均値（バイト）

d：スキーマの認可識別子の長さの平均値（バイト）

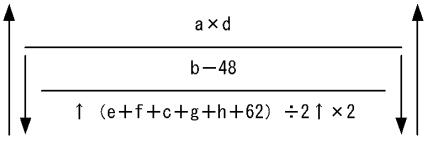
e：総スキーマ総数（個）

f：全 RD エリアを構成する HiRDB ファイルの総数（個）

g：全 RD エリアを構成する HiRDB ファイルの名称の長さの平均値（バイト）

h：パスワード長の平均値（バイト）

### (f) 計算式 6

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_DIV_TABLE_<br>REGULARIZE ※ |  |



a：横分割表の総数（個）

b：データディクショナリ用 RD エリアのページ長（バイト）

c：表を格納する RD エリアの名称長の平均値（バイト）

d：表の横分割条件数の平均値（個）

e：認可識別子の長さの平均値（バイト）

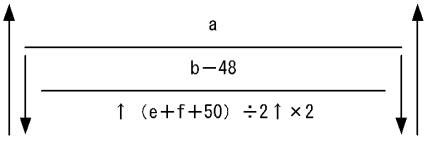
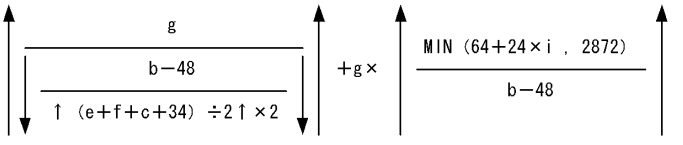
f：表識別子の長さの平均値（バイト）

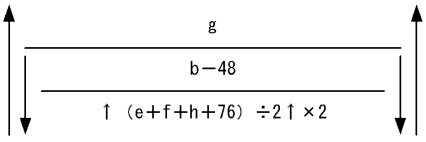
g：表の横分割条件を指定する列が文字列のときの条件値長の平均値（バイト）

h：表の横分割条件を指定する列が数値列のときの条件値長の平均値（バイト）

注※　システムが使用する表です。

### (g) 計算式 7

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_TABLE_<br>STATISTICS ※1 |  |
| SQL_COLUMN_<br>STATISTICS ※1 |  |

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_INDEX_<br>STATISTICS※1 |  |



a：最適化情報を取得する表の総数（個）

b：データディクショナリ用 RD エリアのページ長（バイト）

c：列の名称長の平均値（バイト）

e：認可識別子の長さの平均値（バイト）

f：表識別子の長さの平均値（バイト）

g：最適化情報を取得する表に定義するインデクスの総数（個）

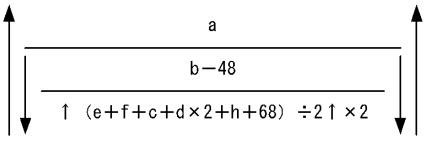
h：インデクス識別子の長さの平均値（バイト）

i：最適化情報を取得する表に定義するインデクスのキー列値数※2（個）

注※1　システムが使用する表です。

注※2　キー列値数 ＜ 100 の場合，i ＝キー列値数
　　キー列値数 ≧ 100 の場合，i ＝ 100

**（h）計算式 8**

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_DIV_<br>COLUMN |  |



a：LOB 列を定義した表の総数（個）

b：データディクショナリ用 RD エリアのページ長（バイト）

c：列の名称長の平均値（バイト）

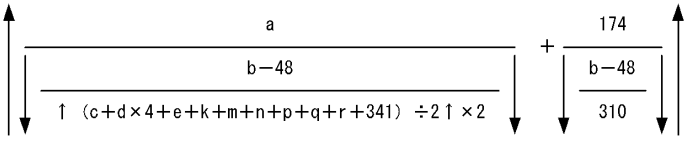
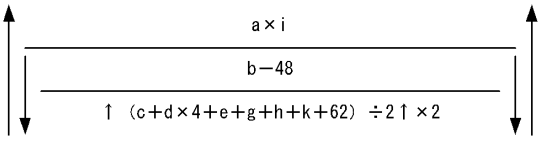
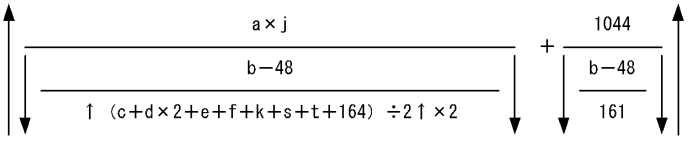
d：表を格納する RD エリアの名称の長さの平均値（バイト）

e：認可識別子の長さの平均値（バイト）

f：表識別子の長さの平均値（バイト）

h：コンポネント名称の長さの平均値（バイト）

**(i) 計算式 9**

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_ROUTINES |  |
| SQL_ROUTINE_ RESOURCES |  |
| SQL_ROUTINE_ PARAMS |  |

a：ルーチンの総数（個）

b：データディクショナリ用 RD エリアのページ長（バイト）

c：ルーチン名称の長さの平均値（バイト）

d：認可識別子の長さの平均値（バイト）

e：特定名※1 の長さの平均値（バイト）

f：パラメタ名称の長さの平均値（バイト）

g：リソース※2 の所有者の認可識別子の長さの平均値（バイト）

h：リソース※2 名称の長さの平均値（バイト）

i：1 ルーチン当たりのリソース※2 数の平均値（個）

j：1 ルーチン当たりのパラメタ数の平均値（個）

k：抽象データ型の名称長の平均値（バイト）

m：ユーザ定義型の名称長（戻り値）の平均値（バイト）

n：外部ルーチン名称長の平均値（バイト）

p：Java クラス名称長の平均値（バイト）

q：Java アーカイブ名称長の平均値（バイト）

r：Java 戻り値のデータ型名称長の平均値（バイト）

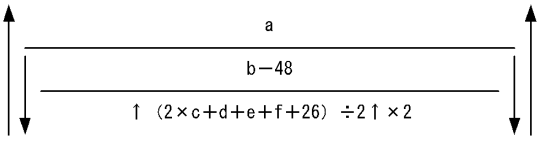
s：Java パラメタデータ型の名称長の平均値（バイト）

t：新旧値相関名で使用した列名称の長さの平均値（バイト）

注※ 1 「認可識別子 . ルーチン識別子」を表しています。

注※ 2 表及びインデクスを表しています。

**(j) 計算式 10**

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_ALIASES |  |



a：別名を定義した表の総数（個）

b：データディクショナリ用 RD エリアのページ長（バイト）

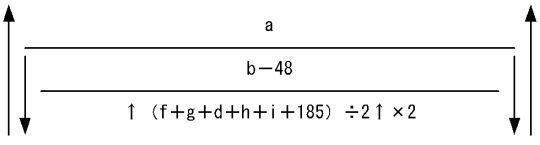
c：別名を定義した表の認可識別子の長さの平均値（バイト）

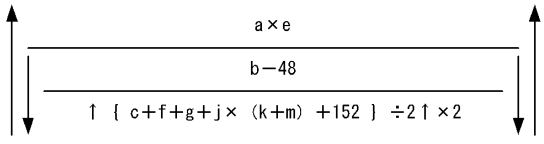
d：表の別名称の長さの平均値（バイト）

e：RD ノード名称の長さの平均値（バイト）

f：基表の名称の長さの平均値（バイト）

**(k) 計算式 11**

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_DATATYPES |  |

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_DATATYPE_<br>DESCRIPTORS |  |



a：ユーザ定義型の総数（個）

b：データディクショナリ用 RD エリアのページ長（バイト）

c：属性又はフィールド名の長さの平均値（バイト）

d：ユーザ定義型の注釈の長さの平均値（バイト）

e：1 データ型当たりの属性数の平均値（個）

f：認可識別子の長さの平均値（バイト）

g：データ型識別子の長さの平均値（バイト）

h：スーパタイプの抽象データ型の認可識別子の長さの平均値（バイト）

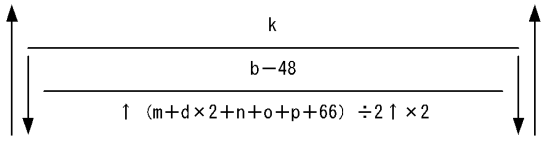
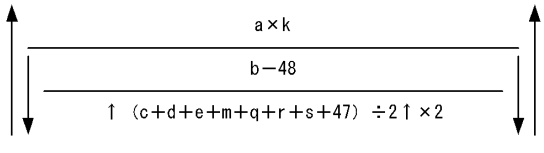
i：スーパタイプの抽象データ型のデータ識別子の長さの平均値（バイト）

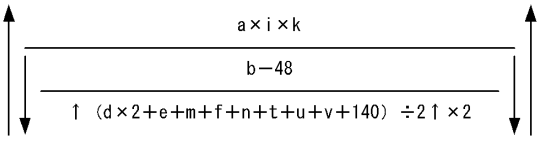
j：ユーザ定義型で定義された属性数（個）

k：ユーザ定義型で定義された属性の抽象データ型の認可識別子の長さの平均値（バイト）

m：ユーザ定義型で定義された属性の抽象データ型のデータ識別子の長さの平均値（バイト）

**(l) 計算式 12**

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_PLUGINS |  |
| SQL_PLUGIN_<br>ROUTINES |  |

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_PLUGIN_<br>ROUTINE_PARAMS |  |



a：1 プラグイン当たりのプラグインルーチン数の平均値（個）

b：データディクショナリ用 RD エリアのページ長（バイト）

c：ルーチン名称の長さの平均値（バイト）

d：認可識別子の長さの平均値（バイト）

e：特定名※の長さの平均値（バイト）

f：1 プラグイン当たりのパラメタ名称の長さの平均値（バイト）

i：1 ルーチン当たりのリソース数の平均値（個）

k：プラグインの総数（個）

m：プラグイン名称の長さの平均値（バイト）

n：抽象データ型／インデクス型名称の長さの平均値（バイト）

o：プラグインライブラリパス名称の長さの平均値（バイト）

p：プラグインの注釈の長さの平均値（バイト）

q：契機指示子の長さの平均値（バイト）

r：オペレーション修飾子の平均値（バイト）

s：オペレーション修飾コード長の平均値（バイト）

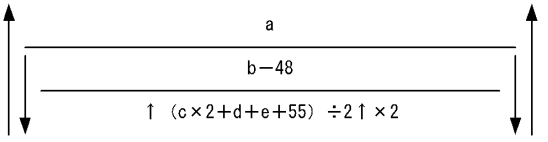
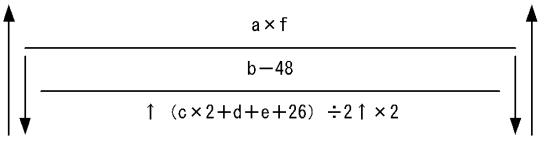
t：パラメタ修飾情報長の平均値（バイト）

u：バインドオペレーション名称長の平均値（バイト）

v：パラメタ修飾情報コード長の平均値（バイト）

注※ 「認可識別子 . ルーチン識別子」を表しています。

**（m）計算式 13**

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_INDEX_TYPES |  |
| SQL_INDEX_TYPE_<br>FUNCTION |  |





a：インデクス型の総数（個）

b：データディクショナリ用 RD エリアのページ長（バイト）

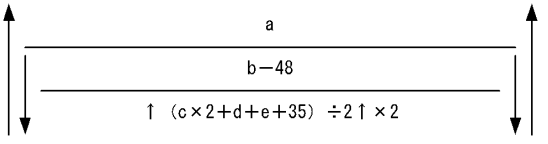
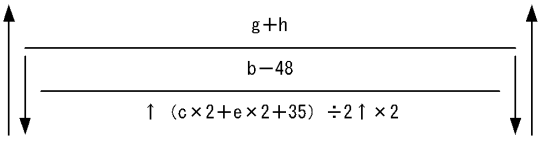
c：認可識別子の長さの平均値（バイト）

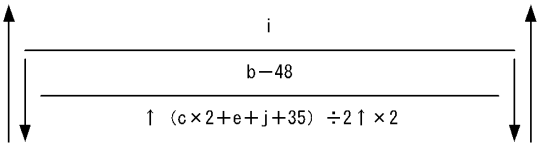
d：インデクス型識別子の長さの平均値（バイト）

e：抽象データ型名称の長さの平均値（バイト）

f：1 インデクス型当たりの適用関数の数（個）

**（n）計算式 14**

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_INDEX_<br>RESOURCES |  |
| SQL_TYPE_<br>RESOURCES |  |

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_TABLE_<br>RESOURCES |  |



a：プラグインインデクスの総数（個）

b：データディクショナリ用 RD エリアのページ長（バイト）

c：認可識別子の長さの平均値（バイト）

d：インデクス型識別子の長さの平均値（バイト）

e：抽象データ型名称の長さの平均値（バイト）

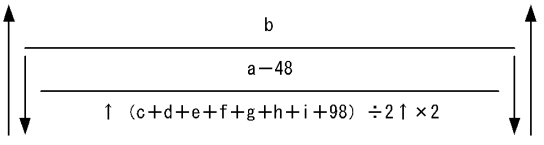
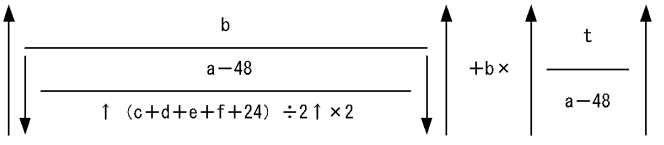
g：抽象データ型で定義された属性の総数（個）

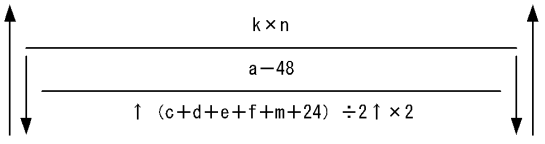
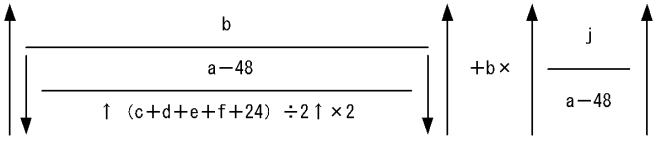
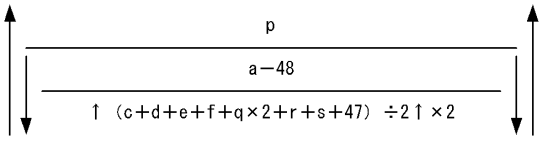
h：サブタイプとして定義された抽象データ型の総数（個）

i：抽象データ型の総数（個）

j：表識別子の平均長（バイト）

**（o）計算式 16**

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_TRIGGERS |  |
| SQL_TRIGGER_ACTCOND<br>※ |  |

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_TRIGGER_COLUMNS |  |
| SQL_TRIGGER_DEF_SOURCE |  |
| SQL_TRIGGER_USAGE |  |

a：データディクショナリ用 RD エリアのページ長（バイト）

b：トリガ定義の総数（個）

c：トリガ認可識別子の長さの平均値（バイト）

d：トリガ名称の長さの平均値（バイト）

e：トリガを定義した表の認可識別子の長さの平均値（バイト）

f：トリガを定義した表名称の長さの平均値（バイト）

g：旧値相関名称の長さの平均値（バイト）

h：新値相関名称の長さの平均値（バイト）

i：トリガ動作手続きの特定名称の長さの平均値（バイト）

j：トリガ定義時の SQL 文の長さの平均値（バイト）

k：UPDATE 文を契機とするトリガの定義数（個）

m：トリガの実行契機となる列に指定した列名称の長さの平均値（バイト）

n；トリガの実行契機となる列に指定した列数の平均値（個）

p：トリガ動作の探索条件中のリソース数（個）

q：トリガ動作の探索条件中のリソースの認可識別子の長さの平均値（バイト）

r：トリガ動作の探索条件中のリソースの表名称の長さの平均値（バイト）

s：トリガ動作の探索条件中のリソースの特定名称の長さの平均値（バイト）

t：トリガ動作条件の解析ツリー長（バイト）

次の計算式で求められる値になります。なお，この計算式中の変数はすべて WHEN の探索条件での指定内容です。

S × 36 ＋ T ＋ U × 48 ＋ V × 128
＋ F1 × 420 ＋ F2 × 132 ＋ F3 × 124 ＋ F4 × 296 ＋ F5 × F4 × 132
＋ A × 140 ＋ B × 200 ＋ 1000

A：コンポーネント指定の属性数（個）

B：値式に現れる抽象データ型の数（個）

F1：システム定義スカラ関数の数（個）

F2：システム定義スカラ関数の引数の数（個）

F3：ユーザ定義関数の数（個）

F4：ユーザ定義関数の候補となる関数の数（個）

F5：ユーザ定義関数の引数の数（個）

S：論理演算子，算術演算子，定数，及びシステム組込みスカラ関数の合計数（個）

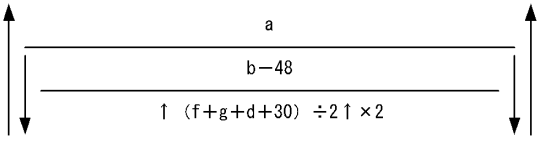
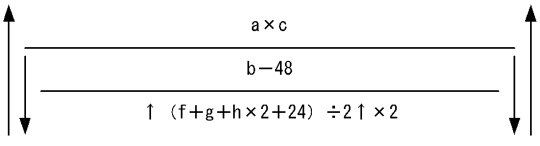
T：定数（HiRDB で解釈するデータ型）の合計長（バイト）

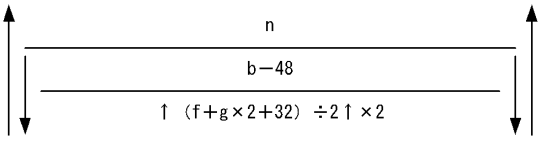
U：スカラ関数 VALUE，CASE 式，CAST 指定中の値式の数（個）

V：列指定の数（個）

注※　システムが使用する表です。

**（p）計算式** 17

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_PARTKEY |  |
| SQL_PARTKEY_DIVISION |  |

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_DIV_TYPE |  |



a：作成するマトリクス分割表の数（個）

b：データディクショナリ用 RD エリアのページ長（バイト）

c：表の横分割条件値数の平均値（個）

d：表の横分割条件を指定する列の名称長の平均値（バイト）

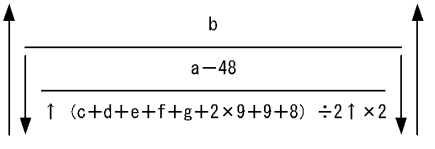
f：認可識別子の長さの平均値（バイト）

g：表識別子の長さの平均値（バイト）

h：表の横分割条件値の長さの平均値（バイト）

n：境界値指定のキーレンジ分割とハッシュ分割を組み合わせたマトリクス分割表の数（個）

**（q）計算式** 18

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_AUDITS |  |



a：データディクショナリ用 RD エリアのページ長（バイト）

b：複製した RD エリアを構成する HiRDB ファイル数（個）

c：イベントタイプ名称の長さの平均値（バイト）

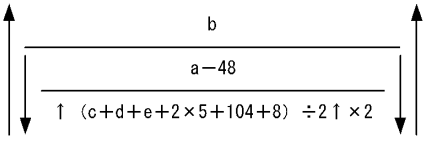
d：イベントサブタイプ名称の長さの平均値（バイト）

e：オブジェクトタイプ名称の長さの平均値（バイト）

f：オブジェクトの所有者名称の長さの平均値（バイト）

g：オブジェクト名称の長さの平均値（バイト）

**(r) 計算式 19**

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_AUDIT_REGULARIZE※ |  |



a：データディクショナリ用 RD エリアのページ長（バイト）

b：複製した RD エリアを構成する HiRDB ファイル数（個）

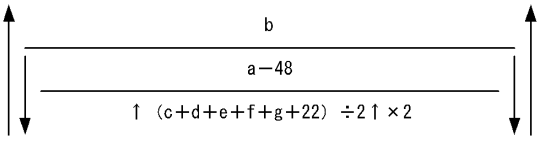
c：オブジェクトタイプ名称の長さの平均値（バイト）

d：オブジェクトの所有者名称の長さの平均値（バイト）

e：オブジェクト名称の長さの平均値（バイト）

注※　システムが使用する表です。

**(s) 計算式 20**

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_KEYCOLUMN_USAGE |  |



a：データディクショナリ用 RD エリアのページ長（バイト）

b：制約定義の総数（個）

c：制約認可識別子の長さの平均値（バイト）

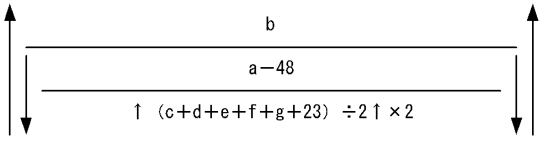
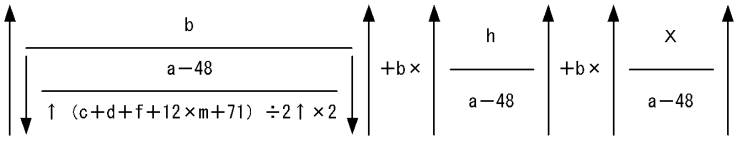
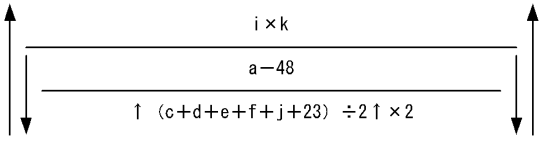
d：制約名称の長さの平均値（バイト）

e：制約を定義した表の認可識別子の長さの平均値（バイト）

f：制約を定義した表名称の長さの平均値（バイト）

g：制約の種類名称の長さの平均値（バイト）

**(t) 計算式 21**

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_TABLE_CONSTRAINTS |  |
| SQL_CHECKS |  |
| SQL_CHECK_COLUMNS |  |

a：データディクショナリ用 RD エリアのページ長（バイト）

b：制約定義の総数（個）

c：制約認可識別子の長さの平均値（バイト）

d：制約名称の長さの平均値（バイト）

e：制約を定義した表の認可識別子の長さの平均値（バイト）

f：制約を定義した表名称の長さの平均値（バイト）

g：制約の種類名称の長さの平均値（バイト）

h：検査制約定義時の SQL 文の長さの平均値（バイト）

i：検査制約の定義数（個）

j：検査制約を定義した列に指定した列名称の長さの平均値（バイト）

k：検査制約を定義した列に指定した列数の平均値（個）

m：バイナリデータ（検査制約の探索条件，検査制約用解析ツリー）のデータ長が 256 以上，かつ分岐する平均バイナリデータ数

バイナリデータの格納ページ数の分岐条件については，「16.1.2　表の格納ページ数の計算方法」を参照してください。

X：検査制約用解析ツリー長（バイト）

次の計算式で求められる値になります。なお，この計算式中の変数はすべて検査制約の探索条件の指定です。

**32 ビットモードの場合：**S × 36 ＋ T ＋（U1 ＋ U2 ＋ U3）× 48 ＋ V × 128 ＋ 1000

**64 ビットモードの場合：**S × 72 ＋ T ＋（U1 ＋ U2 ＋ U3）× 96 ＋ V × 184 ＋ 1400

S：論理演算子，算術演算子（＋，－，＊，／，又は｜｜），及びシステム組み込みスカラ関数の合計数（個）

T：定数（HiRDB で解釈するデータ型）の長さの合計値（バイト）

U1：CASE 式，CAST 指定中の値式の数（個）

U2：スカラ関数（VALUE，SUBSTR，BIT_AND_TEST，POSITION，TIMESTAMP，VARCHAR_FORMAT，TIMESTAMP_FORMAT）の値式の数（個）

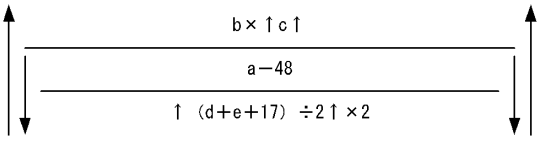
U3：日時書式の数（個）

V：列指定の数（個）

**（u） 計算式** 22

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_SYSPARAMS | 2 |

**（v） 計算式** 23

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_PUBLICVIEW_<br>SAME_USERS |  |

a：データディクショナリ用 RD エリアのページ長（バイト）

b：パブリックビュー表の総定義数（個）

c：パブリックビュー表ごとの重複名数の平均値※（バイト）

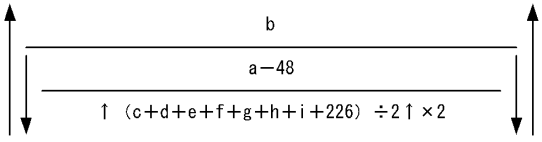
d：パブリックビュー表の表識別子の長さの平均値（バイト）

e：認可識別子の長さの平均値（バイト）

注※

パブリックビュー表の表識別子ごとに SQL_TABLES 表の TABLE_NAME 列が同じ値となる行数の平均値

**（w）計算式 24**

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_SEQUENCES |  |



a：データディクショナリ用 RD エリアのページ長（バイト）

b：順序数生成子の総定義数（個）

c：順序数生成子識別子の長さの平均値（バイト）

d：認可識別子の長さの平均値（バイト）

e：順序数生成子開始値の長さの平均値（バイト）

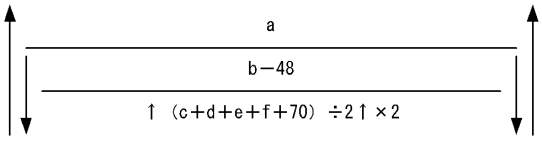
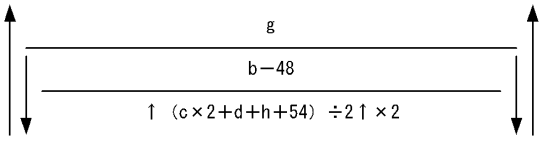
f：順序数生成子最大値の長さの平均値（バイト）

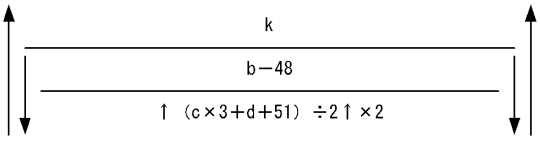
g：順序数生成子最小値の長さの平均値（バイト）

h：順序数生成子増分値の長さの平均値（バイト）

i：RD エリア名称の長さの平均値（バイト）

**（x）計算式 25**

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_FOREIGN_<br>SERVERS |  |
| SQL_USER_<br>MAPPINGS |  |

| ディクショナリ表名 | 計算式 |
|---|---|
| SQL_USAGE_<br>PRIVILEGES ※ |  |

a：外部サーバの総数（個）

b：データディクショナリ用 RD エリアのページ長（バイト）

c：認可識別子の長さの平均値（バイト）

d：外部サーバの名称の長さの平均値（バイト）

e：外部サーバに使用する DBMS の名称の長さの平均値（バイト）

f：外部サーバに使用する DBMS のバージョン識別子の長さの平均値（バイト）

g：ユーザマッピングの総数（個）

h：外部サーバを使用するユーザのパスワードの長さの平均値（バイト）

k：USAGE 権限定義の数（個）
一つの外部サーバに対して n 人に権限を付与した場合，外部サーバ数× n になります。

注※　システムが使用する表です。

## (3) インデクスの格納ページ数の計算方法

インデクスの格納ページ数は，次に示す計算式で求めます。

**計算式**

インデクスの格納ページ数（単位：ページ）
＝ディクショナリ表のインデクスを格納するページ数※＋ 12

注※
「16.1.3　インデクスの格納ページ数の計算方法」を参照し，ディクショナリ表のインデクスを格納するページ数を計算してください。また，このときの計算条件を次に示します。

1. 計算式には，「表 16-6　インデクスの格納ページ数を求める計算式に代入する変数一覧」に示す変数を代入します。
2. 変数 b（CREATE INDEX で指定する未使用領域の比率）は 30 とします。
3. 変数 e 及び f（キー値の重複が dx ＋ 1 以上のキーの種類の個数及びキーの重複数の平均値）は 12 とします。

**表 16-6 インデクスの格納ページ数を求める計算式に代入する変数一覧**

| 表名 | 種別 | キー長※3（変数 g ※1） | キーの種類の個数（変数 c ※1） | キーの重複数の平均値（変数 d ※1） |
|---|---|---|---|---|
| SQL_PHYSICAL_FILES | 1 | 8 | サーバ数 | サーバ内の HiRDB ファイル数の平均値 |
| | 2 | g＋1 | RD エリア数 | RD エリアを構成する HiRDB ファイル数の平均値 |
| SQL_RDAREAS | 3 | g＋1 | RD エリア数 | 1 |
| | 4 | 4 | | |
| SQL_TABLES | 5 | d＋e＋2 | 表の総数＋80 | 1 |
| | 6 | 4 | | |
| SQL_COLUMNS | 7 | d＋e＋f＋3 | a × b | 1 |
| | 8 | d＋e＋6 | | |
| | 9 | 4 | | b |
| SQL_INDEXES | 10 | d＋e＋2 | a | ↑ h ÷ a ↑ |
| | 11 | d＋i＋2 | h | 1 |
| | 12 | 4 | | |
| SQL_USERS | 13 | d＋1 | 認可識別子の数 | 1 |
| SQL_RDAREA_PRIVILEGES | 14 | d＋1 | データベース初期設定ユティリティ（pdinit）の USER USED BY オペランドで指定する認可識別子を重複排除した数 | 1 ユーザ当たりの平均 RD エリア利用権限使用数 |
| | 15 | g＋1 | データベース初期設定ユティリティ（pdinit）の USER USED BY オペランドで指定する RD エリアの数 | 1RD エリア当たりの平均使用ユーザ数 |
| SQL_TABLE_PRIVILEGES | 16 | d＋1 | a | ↑ y ÷ a ↑ |
| | 17 | 2 × d＋e＋3 | y | 1 |
| SQL_DIV_TABLE | 18 | d＋e＋6 | 表の横分割数の総数 | 1 |
| | 19 | g＋1 | 表を横分割したときに指定する RD エリアを重複排除した数 | RD エリアに格納する表数の平均値 |
| | 20 | 4 | | |
| SQL_DIV_TABLE_REGULARIZE | 21 | d＋e＋6 | 表の横分割数の総数 | 1 |
| | 22 | 4 | 横分割表の数 | 表の横分割数の平均値 |
| SQL_INDEX_COLINF | 23 | d＋e＋6 | インデクス定義のある表を重複排除した数 | インデクスを構成する列数の平均値 |

| 表名 | 種別 | キー長※3（変数 g ※1） | キーの種類の個数（変数 c ※1） | キーの重複数の平均値（変数 d ※1） |
|---|---|---|---|---|
| | 24 | d＋i＋6 | インデクスの構成列数 | 1 |
| SQL_TABLE_STATISTICS | 25 | d＋e＋2 | 表の総数（ディクショナリ表もカウントの対象とします） | 1 |
| SQL_COLUMN_STATISTICS | 26 | d＋e＋f＋3 | h | 1 |
| SQL_INDEX_STATISTICS | 27 | d＋e＋2 | インデクスを定義する表の総数 | ↑h÷a↑ |
| | 28 | d＋i＋2 | h | 1 |
| SQL_VIEW_TABLE_USAGE | 29 | d＋e＋2 | z | 1 |
| | 30 | d＋e＋2 | ビュー定義をする基表の数 | 1表当たりのビュー定義数の平均値 |
| | 31 | 4 | z | 1 |
| SQL_VIEWS | 32 | d＋e＋2 | z | 1 |
| | 33 | 4 | | |
| SQL_VIEW_DEF | 34 | d＋e＋2 | z | 1 |
| | 35 | 10 | | |
| SQL_REFERENTIAL_CONSTRAINTS | 39 | d＋e＋2 | 参照制約の総数 | 1 |
| | 40 | d＋e＋2 | 参照表の総数 | 1表当たりの参照表数の平均値 |
| | 41 | d＋e＋2 | 被参照表の総数 | 1表当たりの被参照表数の平均値 |
| SQL_EXCEPT | 86 | d＋e＋2 | 除外値指定をしたインデクスの数 | 一つの表に除外値指定をしたインデクスの数 |
| | 87 | d＋i＋2 | | 1 |
| | 88 | 4 | 除外値指定をしたインデクスを持つ表を重複排除した数 | 一つの表に除外値指定をしたインデクスの数 |
| SQL_DIV_INDEX | 36 | d＋e＋6 | 横分割インデクス数×分割数 | 1 |
| | 37 | d＋i＋2 | 横分割インデクスの総数 | 1表当たりの平均分割数 |
| SQL_DIV_COLUMN | 38 | d＋e＋f＋3 | LOB 列の定義数 | 1表当たりの平均横分割数 |
| | 52 | d＋e＋9 | LOB 属性の定義数 | 1 |
| SQL_ALIASES | 42 | d＋v＋4 | 作成する表別名の総数 | 1 |

| 表名 | 種別 | キー長※3（変数 g※1） | キーの種類の個数（変数 c※1） | キーの重複数の平均値（変数 d※1） |
|---|---|---|---|---|
| SQL_ROUTINES | 43 | d＋MAX（q，7） | p＋174 | 1 |
| | 44 | d＋MAX（t，18） | u＋65 | 1 |
| | 45 | 4 | p＋174 | 1 |
| | 53 | d＋UDT | 抽象データ型の数＋1（NULL 値） | 1 抽象データ型当たりのルーチン数の平均値＋NULL 値数 |
| SQL_ROUTINE_RESOURCES | 46 | d＋q | p×s | 1 ルーチン当たりの使用リソース数の平均値 |
| | 47 | d＋t | | |
| | 48 | d＋q | | |
| | 49 | 4 | | |
| SQL_ROUTINE_PARAMS | 50 | d＋MAX（q，8） | p×r＋347 | 1 ルーチン当たりの平均パラメタ数 |
| | 51 | d＋MAX（t，19） | ルーチン数 | 1 特定名当たりのパラメタ数の平均値（3 未満の場合は 3 としてください） |
| | 106 | e＋4＋2 | トリガ SQL オブジェクト数×r＋1（NULL 値） | 1 特定名当たりのパラメタ数の平均値（3 未満の場合は 3 としてください）＋NULL 値数 |
| SQL_DATATYPES | 54 | d＋UDT | 抽象データ型の数 | 1 |
| | 55 | 4 | | |
| | 56 | d＋UDT | サブタイプを持つ抽象データ型の数＋1（NULL 値） | 1 抽象データ型当たりのサブタイプ数の平均値＋NULL 値数 |
| SQL_DATATYPE_DESCRIPTORS | 57 | d＋UDT＋ATT | NUDT × NATT | 1 |
| | 58 | 4 | 抽象データ型の数 | 1 抽象データ型当たりの属性数の平均値 |
| SQL_TABLE_RESOURCES | 59 | d＋e | ユーザ定義データ型を使用する表の総数 | 1 表当たりの使用ユーザ定義データ型数の平均値 |
| | 60 | d＋UDT | UDT | 1 ユーザ定義データ型当たりの使用表数の平均値 |
| | 61 | 4 | | |

| 表名 | 種別 | キー長※3（変数 g ※1） | キーの種類の個数（変数 c ※1） | キーの重複数の平均値（変数 d ※1） |
|---|---|---|---|---|
| SQL_PLUGINS | 62 | d ＋ PLG | プラグイン数 | 1 |
| | 63 | 4 | | |
| | 64 | {(d ＋ UDT) ÷ IXT} | データ型プラグイン数＋インデクス型プラグイン数 | |
| SQL_PLUGIN_ROUTINES | 65 | t | NPLG × NFPLG | 1 |
| | 66 | PLG ＋ TMD ＋ 2 | | |
| | 67 | PLG ＋ 4 | | |
| | 68 | POPR | オペレーション数 | プラグイン数 |
| SQL_PLUGIN_ROUTINE_PARAMS | 69 | t ＋ PRM | NPLG × NPPAR | 1 |
| | 70 | PLG | NPLG | 1 プラグイン当たりの平均パラメタ数 |
| | 71 | t ＋ 4 | NPLG × NPPAR | 1 |
| SQL_REGISTRY_CONTEXT | 72 | CNM ＋ 1 | コンテキスト数 | 1 |
| SQL_REGISTRY_KEY | 73 | KNM ＋ 6 | キー数 | 1 |
| SQL_INDEX_TYPES | 74 | d ＋ IXT | 作成するインデクス型の数 | 1 |
| | 75 | 4 | | |
| SQL_INDEX_RESOURCES | 76 | d ＋ IXT | プラグインインデクス数 | インデクス型を使用するインデクス定義数の平均値 |
| | 77 | 4 | | |
| SQL_INDEX_DATATYPE | 78 | d ＋ e | プラグインインデクスを定義する表の定義数 | 同一表に対する平均プラグインインデクス数 |
| | 79 | d ＋ i | プラグインインデクス数 | 1 |
| SQL_INDEX_FUNCTION | 80 | d ＋ e | プラグインインデクスを定義する表の定義数 | 1 表当たりのプラグインインデクス数の平均値×プラグインインデクス適用関数数の平均値 |
| | 81 | d ＋ i | プラグインインデクス数 | 1 プラグインインデクス当たりの適用関数数の平均値 |
| SQL_TYPE_RESOURCES | 82 | d ＋ e | ユーザ定義データ型を使用するユーザ定義データ型の数 | ユーザ定義データ型で属性に指定する平均ユーザ定義データ型数 |
| | 83 | d ＋ UDT | ユーザ定義データ型数 | ユーザ定義データ型を使用する平均ユーザ定義データ型数 |
| | 84 | 4 | | |

| 表名 | 種別 | キー長※3（変数 g※1） | キーの種類の個数（変数 c※1） | キーの重複数の平均値（変数 d※1） |
|---|---|---|---|---|
| SQL_INDEX_TYPE_FUNCTION | 85 | d＋IXT | インデクス型の数 | 1 インデクス当たりの適用関数数の平均値 |
| SQL_USER_MAPPINGS | 89 | d＋FSRV | ユーザマッピング数 | 1 |
| SQL_TRIGGERS | 90 | d＋e＋2+16 | トリガ数 | 1 |
| | 91 | d＋TRIG＋2 | | |
| | 92 | d＋t＋2 | | |
| | 93 | 4 | | |
| SQL_TRIGGER_ACTCOND | 94 | d＋TRIG＋2＋4 | 動作条件ありのトリガ数 | 1 |
| | 95 | d＋e＋2 | トリガを定義した表数 | 1 表当たりのトリガ数の平均値 |
| SQL_TRIGGER_COLUMNS | 96 | d＋TRIG＋2 | 列指定がある，UPDATE 文を契機とするトリガ数 | 1 トリガ当たりの指定列の平均値 |
| | 97 | d＋e＋f＋3 | UPDATE 文を契機とするトリガの指定列数 | 1 |
| SQL_TRIGGER_DEF_SOURCE | 98 | d＋TRIG＋2＋4 | トリガ数 | 1 |
| | 99 | d＋e＋2 | トリガを定義した表数 | 1 表当たりのトリガ数の平均値 |
| SQL_TRIGGER_USAGE | 100 | d＋TRIG＋2 | トリガ動作の探索条件中でリソースを参照するトリガ数 | 1 トリガ当たりの参照リソース数の平均値 |
| | 101 | d＋e＋2 | トリガ動作の探索条件中でリソースを参照するトリガを定義した表数 | 1 表当たりの参照リソース数の平均値 |
| | 102 | d＋e＋（t 又は e）＋2 | 使用リソース数 | 1 |
| | 103 | 8 | | |
| SQL_PARTKEY | 104 | d＋e＋f＋3 | マトリクス分割表の数×分割キーの数 | 1 |
| SQL_PARTKEY_DIVISION | 105 | d＋e＋6 | マトリクス分割表の数×境界値数×分割キーの数 | 1 |
| SQL_AUDITS | 107 | ETP＋EST＋2 | 監査対象イベント数 | 1 |

| 表名 | 種別 | キー長※3（変数 g※1） | キーの種類の個数（変数 c※1） | キーの重複数の平均値（変数 d※1） |
|---|---|---|---|---|
| | 108 | OTP＋OSC＋ONM＋3 | 監査対象オブジェクト数 | 1 オブジェクト当たりの監査対象イベント数の平均値 |
| | 109 | d | 監査対象ユーザ数 | 1 ユーザ当たりの監査対象イベント数の平均値 |
| SQL_AUDITS_REGULARIZE | 110 | OTP＋OSC＋ONM＋3 | 監査対象オブジェクト数 | 1 オブジェクト当たりの監査対象イベント数の平均値 |
| | 111 | d | 監査対象ユーザ数 | 1 ユーザ当たりの監査対象イベント数の平均値 |
| SQL_KEYCOLUMN_USAGE | 112 | d＋CNS＋2 | 制約数 | 1 |
| | 113 | d＋e＋2 | 制約を定義している表数 | 1 表当たりの制約数の平均値 |
| SQL_TABLE_CONSTRAINTS | 114 | d＋CNS＋2 | 制約数 | 1 |
| | 115 | d＋e＋2 | 制約を定義した表数 | 1 表当たりの制約数の平均値 |
| SQL_CHECKS | 116 | d＋CNS＋2 | 検査制約数 | 1 |
| | 117 | d＋e＋2 | 検査制約を定義している表数 | 1 表当たりの検査制約数の平均値 |
| SQL_CHECK_COLUMNS | 118 | d＋CNS＋2 | 検査制約数 | 1 検査制約当たりの指定列数の平均値 |
| | 119 | d＋e＋f＋3 | 検査制約で使用している列数 | 1 表当たりの検査制約で使用している列の平均長複数 |
| SQL_SYSPARAMS | 121 | 8 | 2 | 1 |
| SQL_PUBLICVIEW_SAME_USERS | 124 | d＋e＋2 | パブリックビュー表数×認可識別子数 | 1 |
| SQL_INDEX_XMLINF | 125 | d＋e＋2 | NPSIT | 1 インデクス当たりの平均インデクス構成部分構造パス数 |
| | 126 | d＋i＋7 | NPSS | 1 |
| SQL_SEQUENCES | 127 | d＋SEQN＋2 | システム内の順序数生成子数 | 1 |
| | 128 | 4 | システム内の順序数生成子数 | 1 |

a：表の総数（個）

b：表の列数の平均値（個）

c：データディクショナリ用 RD エリアのページ長（バイト）

d：認可識別子の長さの平均値（バイト）

e：表識別子の長さの平均値（バイト）

f：列の名称長の平均値（バイト）

g：RD エリアの名称長の平均値（バイト）

h：インデクスの総数（個）

i：インデクス識別子の長さの平均値（バイト）

n：RD エリアを構成する HiRDB ファイル名称の長さの平均値（バイト）

p：作成するルーチン数（個）

q：ルーチン名称の長さの平均値（バイト）

r：1 ルーチン当たりのパラメタ数の平均値（個）

s：1 ルーチン当たりの使用リソース数の平均値（個）

t：特定名※2の長さの平均値（バイト）

u：特定名※2の総数（個）

v：別名表名称の長さの平均値（バイト）

y：アクセス権限の定義数（個）
一つの表に対して n 人に権限を与えた場合，表数× n 個と数えます。

z：ビュー定義の総数（個）

NUDT：作成するユーザ定義データ型の数（個）

UDT：ユーザ定義データ型名称の長さの平均値（バイト）

NATT：1 ユーザ定義データ型当たりの属性数の平均値（個）

ATT：ユーザ定義データ型属性名称の長さの平均値（バイト）

PLG：プラグイン名称の長さの平均値（バイト）

NPLG：作成するプラグイン数（個）

IXT：インデクス型名称の長さの平均値（バイト）

NFPLG：プラグイン関数の数の平均値（個）

POPR：プラグイン関数名の長さの平均値（バイト）

NPPAR：1 プラグイン関数当たりのパラメタ数の平均値（個）

PRM：1 プラグイン関数当たりのパラメタ名の長さの平均値（バイト）

TMD：契機記述長の平均値（バイト）

CNM：コンテキスト名称長の平均値（バイト）

KNM：キー名称長の平均値（バイト）

FSRV：外部サーバ名称長の平均値（バイト）

TRIG：トリガ名称長の平均値（バイト）

ETP：イベントタイプ名称の長さの平均値（バイト）

EST：イベントサブタイプ名称の長さの平均値（バイト）

OTP：オブジェクト種別名称の長さの平均値（バイト）

OSC：オブジェクトの所有者名称の長さの平均値（バイト）

ONM：オブジェクト名称の長さの平均値（バイト）

CNS：制約名称の長さの平均値（バイト）

NPSIT：部分構造インデクス定義表数（個）

NPSS：部分構造インデクス構成部分構造パス数（個）

SEQN：順序数生成子識別子の長さの平均値（バイト）

注※ 1

「16.1.3　インデクスの格納ページ数の計算方法」の「(2) 計算式中で使用する変数」に記載されている変数のことです。

注※ 2

「認可識別子 . ルーチン識別子」を表しています。

注※ 3

キー長は 4 バイト単位で切り上げになります。次に示す計算式で求めてください。

- ↑キー長÷ 4 ↑× 4

## 16.2.2　解析情報表及び運用履歴表を格納するデータディクショナリ用 RD エリアの容量の見積もり

create rdarea 文に datadictionary of dbmanagement を指定する場合のデータディクショナリ用 RD エリアの容量は，次に示す計算式で求めます。

**計算式**

データディクショナリ用 RD エリアの容量＝（↑（a × 1.3）÷ b ↑）× b × 4096
（単位：バイト）

a：データディクショナリ用 RD エリアの総ページ数※ 1

b：データディクショナリ用 RD エリアのセグメントサイズ※2

注※ 1

「表の格納ページ数＋インデクスの格納ページ数」です。「(1) 表の格納ページ数の計算方法」及び「(2) インデクスの格納ページ数の計算方法」を参照してください。

注※ 2

データベース構成変更ユティリティ（pdmod）の create rdarea 文で指定するセグメントサイズです。

## (1) 表の格納ページ数の計算方法

表の格納ページ数（単位：ページ）は，計算式 1 と計算式 2 の和になります。

| | |
|---|---|
| 計算式 1 | $\uparrow \frac{((a+c+e+g) \times (\uparrow b \uparrow +1) +120) \times 30}{96} \uparrow$ |
| 計算式 2 | $\uparrow \frac{((a+c+e+g) \times (\uparrow b \uparrow +1) +120) \times 30}{9} \uparrow$ |

a：作成する表の数＋ 61（個）

b：表を格納する RD エリアの分割数の平均値（個）

分割していない場合は 1 とします。また，平均値は切り上げます。

c：作成するインデクス数＋ 124（個）

e：作成する表に定義した BLOB 型の列の総数＋ 3（個）

g：作成する表に定義した BLOB 属性の総数（個）

## (2) インデクスの格納ページ数の計算方法

インデクスの格納ページ数（単位：ページ）は次の計算式で求めます。

| |
|---|
| ディクショナリ表のインデクスを格納するページ数※× 2 |

注※

「16.1.3　インデクスの格納ページ数の計算方法」を参照し，ディクショナリ表のインデクスを格納するページ数を計算してください。また，このときの計算条件を次に示します。

1. CREATE INDEX で指定する未使用領域の比率は 0 とします。
2. 計算式には，次に示す変数を代入します。

| キー長<br>（変数 g ※） | キーの種類の個数<br>（変数 c ※） | キーの重複数の平均値<br>（変数 d ※） |
|---|---|---|
| 12 | （a ＋ c ＋ e ＋ g）×（b ＋ 1）＋ 120 | 30 |

a：作成する表の数（個）

b：表を格納する RD エリアの分割数の平均値（個）

c：作成するインデクス数（個）

e：作成する表に定義した BLOB 型の列の総数（個）

g：作成する表に定義した BLOB 属性の総数（個）

なお，解析情報表及び運用履歴表にはユニークインデクスを定義してないため，変数 h を加算する必要はありません。

注※

「16.1.3　インデクスの格納ページ数の計算方法」の「(2) 計算式中で使用する変数」に記載されている変数のことです。

# 16.3　マスタディレクトリ用 RD エリアの容量の見積もり

マスタディレクトリ用 RD エリアの容量は，次に示す計算式で求めます。

**計算式**

マスタディレクトリ用 RD エリアの容量（単位：バイト）
＝｛
↑（a＋2）÷ 800 ↑× 51 ＋↑（b ＋ 120）÷ 6000 ↑× 51 ＋↑（c ＋ 240）÷ 6000 ↑× 51
＋↑（d ＋ 240）÷ 64000 ↑× 51 ＋↑ e ÷ 64000 ↑× 51 ＋ 2 ＋ 6 × n
｝※1 × 4096※2

ａ：データディクショナリ用 RD エリアの総数＋ユーザ用 RD エリアの総数

ｂ：定義する表の総数

ｃ：定義するインデクスの総数

ｄ：定義するビュー表の総数

ｅ：定義するデータ型及びインデクス型の合計

ｎ：マスタディレクトリ用 RD エリアを構成する HiRDB ファイル数

注※１　マスタディレクトリ用 RD エリアの総ページ数です。

注※２　マスタディレクトリ用 RD エリアのページ長です。

# 16.4　データディレクトリ用 RD エリアの容量の見積もり

データディレクトリ用 RD エリアの容量は，次に示す計算式で求めます。

**計算式**

データディレクトリ用 RD エリアの容量（単位：バイト）

$$= \{\uparrow (\sum_{i=1}^{e} gi + \sum_{j=1}^{f} pj + 86) \div 3000 \uparrow \times 51 + 6 \times n + 1\}^{※1} \times 4096^{※2}$$

$$gi = \uparrow (5 \times a_i + 2 \times b_i + 2 \times c_i + 48) \div 32 \uparrow$$

$$pj = \uparrow (d_j + 12) \div 16 \uparrow$$

$a_i$：インデクスを構成する列数

$b_i$：インデクスを格納する RD エリア数

$c_i$：インデクスを定義する表を格納する RD エリア数

$d_j$：表を格納する RD エリア数

e：定義するインデクスの総数

f：横分割する表の総数

n：データディレクトリ用 RD エリアを構成する HiRDB ファイル数

注※ 1　データディレクトリ用 RD エリアの総ページ数です。

注※ 2　データディレクトリ用 RD エリアのページ長です。

# 16.5 データディクショナリ LOB 用 RD エリアの容量の見積もり

## (1) ソース格納用のデータディクショナリ LOB 用 RD エリアの容量の見積もり

ソース格納用のデータディクショナリ LOB 用 RD エリアの容量は，次に示す計算式で求めます。

**計算式**

ソース格納用のデータディクショナリ LOB 用 RD エリアの容量（単位：バイト）

$$
\begin{aligned}
&= \{ \\
&\quad \left[ \sum_{i=1}^{a} \uparrow Si \div 64000 \uparrow \times 96 + 7 + 3 \times (a - 1) \right] \text{※1} \\
&\quad + \left[ \sum_{j=1}^{b} \uparrow (Cj + 1024) \div 8192 \uparrow \right] \text{※2} \\
&\} \text{※3} \times 8192 \text{※4}
\end{aligned}
$$

a：ソース格納用のデータディクショナリ LOB 用 RD エリアを構成する HiRDB ファイル数

b：次の総数

- 手続き（CREATE PROCEDURE）
- 抽象データ型内の関数と手続き（各 FUNCTION（ただし，プラグイン関数を除きます），PROCEDURE）
- ユーザ定義関数（CREATE FUNCTION）

$S_i$：セグメント数

データベース初期設定ユティリティ（pdinit）又はデータベース構成変更ユティリティ（pdmod）の create rdarea 文で指定する各 HiRDB ファイルのセグメント数

$C_j$：次の長さ

- 手続き（各 CREATE PROCEDURE の長さ）
- 抽象データ型内の関数と手続き（各 FUNCTION（ただし，プラグイン関数を除きます），PROCEDURE の長さ）
- ユーザ定義関数（CREATE FUNCTION の長さ）

注※1　ディレクトリページ部分の総ページ数です。

注※2　データページ部分の総ページ数です。

注※3　ソース格納用のデータディクショナリ LOB 用 RD エリアの総ページ数です。

注※4　ソース格納用のデータディクショナリ LOB 用 RD エリアのページ長です。

## (2) オブジェクト格納用のデータディクショナリ LOB 用 RD エリアの容量の見積もり

オブジェクト格納用のデータディクショナリ LOB 用 RD エリアの容量は，次に示す計算式で求めます。

**計算式**

```
オブジェクト格納用のデータディクショナリ LOB 用 RD エリアの容量（単位：バイト）
＝｛
      a
  ［Σ ↑ Si ÷ 64000 ↑× 96 ＋ 7 ＋ 3 ×（a － 1)］※1
     i=1
        b
  ＋［Σ ↑（Cj ＋ 1024）÷ 8192 ↑］※2
       j=1
  ｝※3 × 8192 ※4 ＋ 500000 ※5
```

a：オブジェクト格納用のデータディクショナリ LOB 用 RD エリアを構成する HiRDB ファイル数

b：手続き（CREATE PROCEDURE），抽象データ型内の関数と手続き（各 FUNCTION（ただし，プラグイン関数を除きます），PROCEDURE），ユーザ定義関数（CREATE FUNCTION）及びトリガ定義（CREATE TRIGGER）の総数

$S_i$：セグメント数

データベース初期設定ユティリティ（pdinit）又はデータベース構成変更ユティリティ（pdmod）の create rdarea 文で指定する各 HiRDB ファイルのセグメント数

$C_j$：$QO_i$ ＋ PR（$QO_i$ の計算式を（3）に，PR の計算式を（4）に示します。これらの計算式で使用する変数を（5）に示します）

注※1　ディレクトリページ部分の総ページ数です。

注※2　データページ部分の総ページ数です。

注※3　オブジェクト格納用のデータディクショナリ LOB 用 RD エリアの総ページ数です。

注※4　オブジェクト格納用のデータディクショナリ LOB 用 RD エリアのページ長です。

注※5　抽象データ型，又はプラグイン使用時に加算します。

### （3）$QO_i$（SQL オブジェクト長）の計算式

```
QOi（単位：バイト）＝
 a
Σ ｛
i=1
    1840 ＋ 46 × RCN ＋ 298 × Si ＋ 20 × Pi ＋ 1138 × Ti ＋ 76 × Ti × Di ＋ 80 × Ci ＋ 40 × Ii ＋ 534 × Wi
    ＋ 20 × Ki ＋ Li ＋ 8 × TCi ＋ 656 × Di ＋ 48 × nFF ＋ 100 × nFP ＋ 148 × nFC ＋ 696 × nPFF
    ＋ 16 ×（nAT ＋ nPAT）＋ 20 × nCAT ＋ 28 ×（nAF ＋ nCAF）＋ 20 ×（nAA ＋ nPAA ＋ nCAA）
    ＋ 1057 × nSPA ＋ 120 × nSPP ＋ 287 × nSFF ＋ 8 × nSFP ＋ 813 × nJFC ＋ 20 × nJFP
    〔＋ 60 × nFT ＋ nFTS ＋ 32 × nFTQ〕※1
    〔＋ 1057 × nTR ＋ 120 ×（nTSN ＋ nTSO）＋ 20 ×（nTCN ＋ nTCO)〕※2
    〔＋ 760 ＋ 376 × RCC ＋ 1880 × RCT〕※3
    〔＋ 32 × Si ＋ 16〕※4
    〔＋↑（42 × SiT）＋｛52 ＋ 152 ×（SiTA ＋ SiSA ＋ SiNA）×（SiT ＋ SiS ＋ SiN)｝↑〕※5
  ｝
```

a：ストアドプロシジャ内の SQL 文数

注※1　外部表を指定する場合に加算する計算式です。

注※2　トリガを使用する場合に加算する計算式です。

注※3　参照制約を使用する場合に加算する計算式です。

注※4　列名記述領域長の計算式です。動的 SQL の場合に加算します。

注※5　型名記述領域長の計算式です。動的 SQL の場合に加算します。

## (4) PR（ルーチン制御用オブジェクト長）の計算式

### (a) ユーザが定義する場合

ストアドプロシジャ，ストアドファンクション，又はトリガを定義した場合の，ルーチン制御用オブジェクト長は次に示す計算式から求めます。

```
PR（単位：バイト）＝
 a
Σ {
i=1
    600 ＋ 28 × sRi ＋ 32 ×（sRUi ＋ sDi）＋ 56 × sSXi ＋ sCUi ＋ sSi ＋ sPi ＋ sLA
    ＋ sKi ＋ sL ＋ 80 × sWi ＋ 24 × sCM ＋ 32 × sCCR ＋ 2 × sDCR ＋ 60 × sCHD ＋ 72 × sDHD ＋ 64 × sHCN
    ＋ 8 × sCHD × sHCN ＋ 48 × nRFF ＋ 100 × nRFP ＋ 148 × nRFC ＋ 200 × nPRFF ＋ 8 × nPRFP
    ＋ 196 × nPA ＋ 64 × nPP ＋ 36 × nPPI ＋ 20 × nPPO ＋ 200 × nPPA ＋ 8 × nPPP ＋ 20 × nAR ＋ 48 × nARA
    ＋ 16 × nRPAT ＋ 20 × nCAT ＋ 28 ×（nRPAF ＋ nRCAF）＋ 20 ×（nRPAA ＋ nRCAA）＋ 287 × nRSFF
    ＋ 8 × nRSFP ＋ 813 × nPJA ＋ 20 × nPJP ＋ 813 × nRJFC ＋ 20 × nRJFP
    〔＋ 28 ×（nTSN × 2 ＋ nTSO)〕※
  }
```

a：次の SQL 文数

- 手続き（CREATE PROCEDURE）
- 抽象データ型内の関数と手続き（各 FUNCTION（ただし，プラグイン関数を除きます），PROCEDURE）
- ユーザ定義関数（CREATE FUNCTION）
- トリガ定義（CREATE TRIGGER）

注※　トリガを使用する場合に加算する計算式です。

### (b) HiRDB が自動的に作成する場合

表定義時，参照動作に CASCADE を指定した場合，HiRDB が制約制御のためにトリガを作成したときのルーチン制御用オブジェクト長は次に示す計算式から求めます。

```
PR（単位：バイト）＝
 a
Σ {240 ＋ 608 × RCC ＋（5120 ＋ 100 × RDi ＋ 256 × RIi）× RCP × RCT}
i=1
```

a：次の SQL 文数

- 手続き（CREATE PROCEDURE）
- 抽象データ型内の関数と手続き（各 FUNCTION（ただし，プラグイン関数を除きます），PROCEDURE）
- ユーザ定義関数（CREATE FUNCTION）
- トリガ定義（CREATE TRIGGER）

### (5) PR 及び $QO_i$ の計算式で使用する変数

| 変数名 | 説明 |
|---|---|
| RCN | SQL オブジェクトで使用する表，及びインデクスの合計数 |
| Si | SQL 文中にある検索項目数（SQL 文で指定する列がインデクス列の場合はその列数） |
| Pi | SQL 文中にある埋込み変数又はパラメタ数 |
| Ti | SQL 文中にある表名数 |
| Ci | SQL 文中にある列名数 |
| TCi | SQL 文中にある表の構成列数 |
| Wi | SQL 文中にある論理演算子数※1 |
| Ki | SQL 文中にある定数の数※1 |
| Li | SQL 文中にある定数の合計長※1（単位：バイト） |
| Ii | SQL 文実行時に使用するインデクス数（SQL 文で指定する表のうち，検索条件に指定するインデクス数） |
| Di | SQL 文中の表に定義された格納条件の総数（マトリクス分割表は 2 倍する） |
| SiT | SQL 文中にある選択式の抽象データ型数 |
| SiS | SQL 文中にある選択式の抽象データ型のスーパタイプ数 |
| SiN | SQL 文中にある選択式のサブタイプである抽象データ型のスーパタイプの総数 |
| SiTA | SQL 文中にある選択式の抽象データ型の属性数 |
| SiSA | SQL 文中にある選択式の抽象データ型のスーパタイプの属性数 |
| SiNA | SQL 文中にある選択式のサブタイプである抽象データ型のコンポネント指定総数 |
| nSPA | SQL 文中にある手続き文の呼び出し数 |
| nSPP | SQL 文中にある手続き文の引数の総数 |
| nFF | SQL 文中にある関数の呼び出し数※1 |
| nFP | SQL 文中にある関数の引数の総数※1 |
| nFC | SQL 文中にある関数の総関数定義候補数（関数呼出し数 nFF に，引数が抽象データ型の各関数呼出しに対して，サブタイプを引数とする関数定義数を加算する） |
| nPFF | SQL オブジェクトが使用するプラグイン関数呼出し数（SQL 文中にあるプラグイン関数呼出し数＋SELECT の場合は 1，INSERT，UPDATE，DELETE の場合は 6） |
| nSFF | SQL 文中にあるシステム定義スカラ関数の呼び出し数※1 |
| nSFP | SQL 文中にあるシステム定義スカラ関数の引数の総数※1 |
| nJFC | SQL 文中にある外部 Java 関数の呼び出し数 |
| nJFP | SQL 文中にある外部 Java 関数の引数の総数 |
| nAT | SQL 文中にあるコンポネント指定で使用する抽象データ型数（スーパタイプ，抽象データ型属性によって現れる抽象データ型を除く） |
| nAA | SQL 文中にあるコンポネント指定で使用する抽象データ型数（スーパタイプ，抽象データ型属性によって現れる抽象データ型を含む） |
| nAF | SQL 文中にあるコンポネント指定で使用する属性総数 |
| nPAT | SQL オブジェクトが使用するプラグイン関数の引数で使用する抽象データ型数（スーパタイプ，抽象データ型属性によって現れる抽象データ型を除く） |
| nPAA | SQL オブジェクトが使用するプラグイン関数の引数で使用する抽象データ型数（スーパタイプ，サブタイプを含む） |

| 変数名 | 説明 |
|---|---|
| nCAT | SQL 文中にあるコンストラクタ関数の呼び出し数 |
| nCAA | SQL 文中にあるコンストラクタ関数の抽象データ型数（スーパタイプを含む） |
| nCAF | SQL 文中にあるコンストラクタ関数の抽象データ型の属性総数 |
| nFT | SQL 文中にある外部表数 |
| nFTS | 外部表を検索する SQL 文の長さ※2 |
| nFTQ | 外部表を検索する SQL 文中の埋込み変数又は？パラメタの数※2 |
| nTR | SQL 文の実行によって起動されるトリガ数 |
| nTSN | SQL 文の実行によって起動される各トリガのトリガ SQL 文中の新値相関名によって修飾された列の総数 |
| nTSO | SQL 文の実行によって起動される各トリガのトリガ SQL 文中の旧値相関名によって修飾された列の総数 |
| nTCN | SQL 文の実行によって起動される各トリガのトリガ動作条件中の新値相関名によって修飾された列の総数 |
| nTCO | SQL 文の実行によって起動される各トリガのトリガ動作条件中の旧値相関名によって修飾された列の総数 |
| RCC | SQL 文中で，更新対象の表を参照する表の外部キーの構成列数と，主キーの構成列数の総数 |
| RCT | SQL 文中で，更新対象の表を参照する表，及び更新対象の表が参照する表の総数 |
| RCP | 参照表定義時に，参照動作に指定した CASCADE の総数 |
| RIi | 参照表定義時，参照指定をする被参照表に定義されたインデクスの総数 |
| RDi | 参照表定義時，参照指定をする被参照表に定義された分割格納条件の総数（マトリクス分割表の場合は，2 倍する） |
| sRi | 手続き及び関数中の SQL パラメタ数（INOUT 指定の SQL パラメタ数は 2 倍する） |
| sRUi | 手続き及び関数中の SQL パラメタの総数（又は，トリガ定義中のトリガ SQL 文中の新旧値相関名によって修飾された列の総数） |
| sDi | 手続き，関数，及びトリガ SQL 文中の SQL 変数（declare）の総数 |
| sSXi | 手続き，関数，及びトリガ SQL 文中の SQLCODE，SQLCOUNT 変数の総数 |
| sCUi | 手続き，関数，及びトリガ SQL 文中の CURRENT_TIME，CURRENT_DATE 定数の総数 |
| sSi | 手続き及びトリガ SQL 文中の操作系 SQL の数（カーソル宣言を除く SQL 文：OPEN，FETCH，CLOSE，UPDATE，DELETE，INSERT 文など） |
| sPi | 手続き，関数，及びトリガ SQL 文中のルーチン制御 SQL 数（BEGIN，SET，IF，ELSEIF，WHILE など） |
| sLA | 手続き，関数，及びトリガ SQL 文中のラベル数 |
| sKi | 手続き，関数，及びトリガ SQL 文中の定数の数（手続き及びトリガ SQL 文中に記述された操作系 SQL 文の定数を除く） |
| sL | 手続き，関数，及びトリガ SQL 文中の定数の長さの合計（手続き及びトリガ SQL 文中に記述された操作系 SQL 文の定数を除く） |
| sWi | 手続き，関数，及びトリガ SQL 文中の条件述語数 |
| sCM | 手続き，関数，及びトリガ SQL 文中の複合文の数 |
| sCCR | 手続き及びトリガ SQL 文のカーソル宣言を記述した複合文の数 |
| sDCR | 手続き及びトリガ SQL 文のカーソル宣言の数 |
| sCHD | 手続き，関数，及びトリガ SQL 文のハンドラ宣言を記述した複合文の数 |
| sDHD | 手続き，関数，及びトリガ SQL 文のハンドラ宣言の数 |
| sHCN | 手続き，関数，及びトリガ SQL 文のハンドラ宣言中に記述された条件値の数 |
| nRFF | ルーチン中にある関数呼出し数 |

| 変数名 | 説明 |
|---|---|
| nRFP | ルーチン中にある関数の引数の総数 |
| nRFC | ルーチン中にある関数の関数定義候補の総数（関数呼出し数 nFF に，引数が抽象データ型の各関数呼出しに対して，サブタイプを引数とする関数定義数を加算する） |
| nPRFF | ルーチンの SQL オブジェクトが使用するプラグイン関数呼出し数 |
| nPRFP | ルーチンの SQL オブジェクトが使用するプラグイン関数のプラグインパラメタ総数 |
| nPA | ルーチン中にある手続き呼び出し数 |
| nPP | ルーチン中にある手続きのパラメタ総数 |
| nPPI | ルーチン中にある手続きの入力パラメタ総数（入出力パラメタを含む） |
| nPPO | ルーチン中にある手続きの出力パラメタ総数（入出力パラメタを含む） |
| nPPA | ルーチンの SQL オブジェクトが使用するプラグイン手続き呼び出し数 |
| nPPP | ルーチンの SQL オブジェクトが使用するプラグイン手続きのプラグインパラメタ総数 |
| nRSFF | ルーチン中にあるシステム定義スカラ関数の呼び出し数 |
| nRSFP | ルーチン中にあるシステム定義スカラ関数の引数の総数 |
| nPJA | ルーチン中にある外部 Java 手続きの呼び出し数 |
| nPJP | ルーチン中にある外部 Java 手続きの引数の総数 |
| nRJFC | ルーチン中にある外部 Java 関数の呼び出し数 |
| nRJFP | ルーチン中にある外部 Java 関数の引数の総数 |
| nAR | ルーチン中にあるコンポネント指定で使用する抽象データ型数（スーパタイプ，抽象データ型属性によって現れる抽象データ型を除く） |
| nARA | ルーチン中にあるコンポネント指定で使用する属性総数 |
| nRPAT | ルーチンの SQL オブジェクトが使用するプラグインルーチンのパラメタで使用する抽象データ型数（ただし，スーパタイプ，抽象データ型属性の抽象データ型を除く） |
| nRPAA | ルーチンの SQL オブジェクトが使用するプラグインルーチンのパラメタで使用する抽象データ型数（スーパタイプを含む） |
| nRPAF | ルーチンの SQL オブジェクトが使用するプラグインルーチンのパラメタで使用する抽象データ型の属性総数 |
| nRCAT | ルーチン中にあるコンストラクタ関数の呼び出し数 |
| nRCAA | ルーチン中にあるコンストラクタ関数の抽象データ型数（スーパタイプを含む） |
| nRCAF | ルーチン中にあるコンストラクタ関数の抽象データ型の属性総数 |

注※ 1

トリガを使用する場合は，SQL 文の実行によって起動される各トリガのトリガ動作条件についても数える必要があります。

注※ 2

外部表を検索する SQL 文はアクセスパス表示ユティリティ（pdvwopt コマンド）で確認できます。

# 16.6 ユーザ LOB 用 RD エリアの容量の見積もり

ユーザ LOB 用 RD エリアの容量は，次に示す計算式で求めます。

**計算式**

ユーザ LOB 用 RD エリアの容量（単位：バイト）
＝（ディレクトリページ部分の総ページ数＋データページ部分の総ページ数）× 8192 ※

注※　ユーザ LOB 用 RD エリアのページ長です。

## (1) ディレクトリページ部分の総ページ数

**計算式**

ディレクトリページ部分の総ページ数

$$= \sum_{i=1}^{a} \uparrow Si \div 64000 \uparrow \times 96 + 7 + 3 \times (a - 1)$$

a：ユーザ LOB 用 RD エリアを構成する HiRDB ファイル数

$S_i$：セグメント数

データベース初期設定ユティリティ（pdinit）又はデータベース構成変更ユティリティ（pdmod）の create rdarea 文で指定する各 HiRDB ファイルのセグメント数

## (2) データページ部分の総ページ数

**計算式**

データページ部分の総ページ数

$$= \sum_{j=1}^{b} \uparrow (Cj + 1024) \div 8192 \uparrow$$

b：LOB 列の行の総数

データ長が 0 の行もカウントします。NULL 値の行はカウントしません。

$C_j$：各 BLOB データのデータ長（バイト）

# 16.7　レジストリ用 RD エリアの容量の見積もり

レジストリ用 RD エリアの容量は，次に示す計算式で求めます。

**計算式**

```
レジストリ用 RD エリアの容量（単位：バイト）
  ＝レジストリ用 RD エリアのページ長※×レジストリ用 RD エリアの総ページ数× 1.3
```

注※

レジストリ機能初期設定ユティリティ（pdreginit）の create rdarea 文で指定するページ長です。

**レジストリ用 RD エリアの総ページ数の求め方**

```
レジストリ用 RD エリアの総ページ数（単位：ページ）
     a
  ＝ Σ  {↑ Si ÷ d ↑＋↑ Si ÷ e ↑＋ 6 ×（a ＋ 1）＋ 2 ×↑ 20480 ÷ b ↑
    i=1
  ＋レジストリ管理表の格納ページ数＋レジストリ管理表のインデクス格納ページ数
```

a：レジストリ用 RD エリアを構成する HiRDB ファイル数

b：レジストリ用 RD エリアのページ長（単位：バイト）

c：レジストリ機能初期設定ユティリティ（pdreginit）の create rdarea 文で指定するセグメントサイズ

d：↓（b － 20）÷ {(↑ c ÷ 32 ↑× 8）＋ 56} ↓

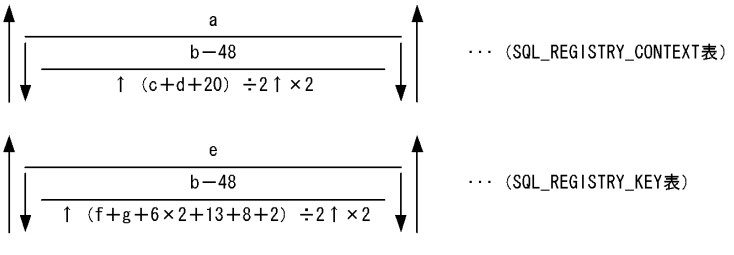
e：↓（125 × b）÷（16 × d）↓× d

Si：レジストリ機能初期設定ユティリティ（pdreginit）の create rdarea 文で指定する各 HiRDB ファイルのセグメント数

各表及びインデクスの確保単位は，セグメントです。表やインデクスごとに求めた値をセグメント単位に切り上げます。

## (1) レジストリ管理表の格納ページ数

**計算式**




a：レジストリ管理表のコンテキスト数

b：レジストリ管理表のページ長

c：レジストリコンテキスト名称の長さ

d：アクセスパスワードの長さ

e：レジストリ管理表のキー値数（レジストリ管理表に登録したキー名称の数）

f：レジストリキー名称長

g：レジストリキー値長（レジストリキー値長が 32,000 バイト以下の場合に加算する）

## (2) レジストリ管理表のインデクス格納ページ数

**計算式**

```
レジストリ管理表のインデクス格納ページ数（単位：ページ）
  ＝ SQL_REGISTRY_CONTEXT 表のインデクス格納ページ数
  ＋ SQL_REGISTRY_KEY 表のインデクス格納ページ数
```

SQL_REGISTRY_CONTEXT 表のインデクス格納ページ数，及び SQL_REGISTRY_KEY 表のインデクス格納ページ数については，「16.1.3　インデクスの格納ページ数の計算方法」を参照してください。ただし，CREATE INDEX 文で指定する未使用領域の比率は 30%として計算してください。

インデクスを格納するページ数の計算式に使用する値を次に示します。

**レジストリ用 RD エリアのインデクス格納ページ数の計算式に使用する値**

| 表名 | 種別 | キー長 | キーの種類 | 平均重複数 |
|---|---|---|---|---|
| SQL_REGISTRY_CONTEXT | 72 | a ＋ 1 | コンテキスト名の数（レジストリ管理表のコンテキスト数） | 1 |
| SQL_REGISTRY_KEY | 73 | f ＋ 6 | キー値の数（レジストリ管理表のキー値数） | 1 |

a：レジストリコンテキストの長さ

f：レジストリキー名称の長さ

# 16.8 レジストリ LOB 用 RD エリアの容量の見積もり

レジストリ LOB 用 RD エリアの容量は，次に示す計算式で求めます。

**計算式**

レジストリ LOB 用 RD エリアの容量（単位：バイト）
＝（ディレクトリページ部分の総ページ数＋データページ部分の総ページ数）× 8192 ※

注※　レジストリ LOB 用 RD エリアのページ長です。

## (1) ディレクトリページ部分の総ページ数

**計算式**

ディレクトリページ部分の総ページ数

$$= \sum_{i=1}^{a} \uparrow Si \div 64000 \uparrow \times 96 + 7 + 3 \times (a - 1)$$

a：レジストリ LOB 用 RD エリアを構成する HiRDB ファイル数

$S_i$：レジストリ LOB 用 RD エリアのセグメント数

## (2) データページ部分の総ページ数

**計算式**

データページ部分の総ページ数

$$= \sum_{j=1}^{b} \uparrow (Cj + 1024) \div 8192 \uparrow$$

b：32000 バイトを超えるレジストリキー値の数

$C_j$：32000 バイトを超えるレジストリキー値の長さ

# 16.9 リスト用 RD エリアの容量の見積もり

リスト用 RD エリアの容量は，次に示す計算式で求めます。

**計算式**

```
リスト用 RD エリアの容量（単位：バイト）
＝｛
    a
    Σ（↑ Si ÷ f ↑＋↑ Si ÷ g ↑）
   i=1
   ＋ 6 ×（a ＋ 1）＋↑（1024 × n）÷（25 × b）↑＋↑ 20480 ÷ b ↑＋ c × e
   ｝× b
```

a：リスト用 RD エリアを構成する HiRDB ファイル数

b：リスト用 RD エリアのページ長※（バイト）

c：リスト用 RD エリアのセグメント数

e：リスト用 RD エリアのセグメントサイズ※

f：↓｛b － 20｝÷｛(↑ e ÷ 32 ↑× 8）＋ 56｝↓

g：↑（125 × b）÷（16 × f）↑× f

n：1 リスト用 RD エリアに作成できる最大リスト数※

リスト用 RD エリアの最大リスト数は，次の計算式で求めます。

↑サーバ内に保持するリストの総数÷サーバ内に配置するリスト用 RD エリアの数↑

$S_i$：リスト用 RD エリアを構成する HiRDB ファイルのセグメント数※

注※

上記の計算式で求めた値以上となる最も小さい 500 の整数倍の値を，データベース初期設定ユティリティ（pdinit）若しくはデータベース構成変更ユティリティ（pdmod）の create rdarea 文，又はデータベース構成変更ユティリティの initialize rdarea 文で指定してください。

# 17 システムファイル及び監査証跡ファイルの容量の見積もり

この章では，システムログファイル，シンクポイントダンプファイル，ステータスファイルなどのシステムファイルの容量の見積もり方法，及び監査証跡ファイルの容量の見積もり方法について説明します。


17.1 **システムログファイルの容量の見積もり**

17.2 **シンクポイントダンプファイルの容量の見積もり**

17.3 **ステータスファイルの容量の見積もり**

17.4 **監査証跡ファイルの容量の見積もり**

# 17.1 システムログファイルの容量の見積もり

ここでは，システムログファイルの容量の見積もり方法について説明します。説明する項目を次に示します。

- システムログファイルの総容量
- 表定義時に出力されるシステムログ量
- インデクス定義時に出力されるシステムログ量
- 表データ更新時に出力されるシステムログ量
- ユティリティによるデータベース作成時に出力されるシステムログ量
- SQL 操作に応じて出力されるシステムログ量
- RD エリアの自動増分機能使用時に出力されるシステムログ量
- PURGE TABLE 文実行時に出力されるシステムログ量
- 空きページ解放ユティリティ実行時に出力されるシステムログ量
- 再編成時期予測機能使用時に出力されるシステムログ量

## 17.1.1 システムログファイルの総容量

### (1) システムログファイルの総容量の求め方

システムログファイルの総容量は，次に示す計算式で求めます。

**計算式**

全システムログファイルの総容量（単位：バイト）＝（↑ a ÷ c ↑× 3）※× b

a：出力されるシステムログ量
求め方については，「(2) システムログ量の求め方」を参照してください。

b：pd_log_rec_leng オペランドで指定するシステムログファイルのレコード長です。

c：次に示す値を代入してください。
pd_log_rec_leng = 1024 の場合：1000
pd_log_rec_leng = 2048 の場合：2000
pd_log_rec_leng = 4096 の場合：4000

注
- システムログファイル中に空き領域が発生するため，ここで求めた容量の 1.2 倍以上の容量を準備することをお勧めします。
- ここで得られる総容量を単位時間当たりの容量に変換して，一つのシステムログファイルの容量と数を見積もってください。また，容量と数を見積もるときに，システムログファイルのアンロード間隔についても考慮してください。

注※
システムログファイルの総レコード数を求める計算式です。

### (2) システムログ量の求め方

出力されるシステムログ量は，次に示す計算式で求めます。

**計算式**

出力されるシステムログ量（単位：バイト）＝ Σ｛b ＋ e ＋↑ b ÷（a － 256）↑× 256 ＋ d｝＋ c
**●回復不要 FES の場合だけ加算します。**
＋ f

Σ はトランザクションごとの合計を意味します。ただし，実行するトランザクションが次の条件をすべて満たす場合，その分のシステムログは出力されません。

- 検索をするトランザクションで，かつ COMMIT 文で決着する
- トランザクションを実行するサーバがシングルサーバ，ディクショナリサーバ，又はバックエンドサーバである

a：pd_log_max_data_size オペランドの値

b：1336 ＋データベース操作に応じて出力されるシステムログ量
ただし，回復不要 FES の場合は 0 になります。
データベース操作に応じて出力されるシステムログ量は，「17.1.2　表定義時に出力されるシステムログ量」～「17.1.5　ユティリティによるデータベース作成時に出力されるシステムログ量」で求めたシステムログ量を一つのトランザクション当たりで合計した値です。

c：SQL 操作に応じて出力されるシステムログ量
「17.1.6　SQL 操作に応じて出力されるシステムログ量」を参照してください。

d：サーバ種別ごとに次に示す計算式で算出した値

- HiRDB/ シングルサーバの場合
  2 × pd_log_rec_leng オペランドの値
- HiRDB/ パラレルサーバの場合
  ・フロントエンドサーバのとき

| マルチスレッド対応で，X/Open に準拠した接続方式を実現する HiRDB ライブラリを使用した UAP の実行，かつトランザクションの移行機能の使用有無 | 回復不要 FES の使用有無 | 計算式 |
|---|---|---|
| 使用 | 未使用（使用できない） | （↑ 1336 ÷ pd_log_rec_leng オペランドの値↑＋ 3）× pd_log_rec_leng オペランドの値 |
| 未使用 | 使用 | 0 |
| 未使用 | 未使用 | 2 × pd_log_rec_leng オペランドの値 |

・バックエンドサーバ又はディクショナリサーバのとき

| システム内の，回復不要 FES を使用するフロントエンドサーバの有無 | pd_rpl_reflect_mode オペランドの指定値 | 計算式 |
|---|---|---|
| あり | uap | pd_log_rec_leng オペランドの値＋↑｛176 ＋ 128 ×（一つのトランザクションで更新する BES の最大数＋ 1 ）｝÷ pd_log_rec_leng オペランドの値↑× pd_log_rec_leng オペランドの値 |
| | server | |
| なし | uap | |
| | server | 2 × pd_log_rec_leng オペランドの値 |

e：

- HiRDB/ パラレルサーバの場合
  フロントエンドサーバのとき：8 ＋ 72 ×一つのトランザクションで参照，又は更新する BES 及び DS の最大数
  フロントエンドサーバ以外のとき：0
- HiRDB/ シングルサーバの場合：0

f：10 × pd_log_rec_leng の値

注　システムログは次に示す操作をする場合に出力されます。

- 表の定義時
- インデクスの定義時
- 表データの更新時
- ユティリティによるデータベースの作成時

なお，上記の操作によってデータベースを更新中に，ロールバックが発生した場合，それまでデータベースを更新した分のシステムログが新たに追加で出力されます。このことを考慮してシステムログ量を見積もってください。

### (3) 容量見積もり時の注意事項

次の SQL を実行すると，システムログを大量に出力します。システムログファイルの容量が不足するおそれがあるため，これらの SQL は出力するシステムログ量を見積もってから実行してください。また，見積もりの結果，システムログファイルの容量が不足する場合は，次に示す方法で容量不足を回避してください。

| システムログを大量に出力する SQL | システムログファイル容量不足の回避方法 |
|---|---|
| CREATE INDEX　形式 1（インデクス定義） | インデクスオプションに EMPTY を指定してインデクスを定義し，データベース再編成ユティリティ（pdrorg）を使用してインデクスを作成します。そのとき，ログ取得方式に更新前ログ取得モードを指定してインデクスを作成してください。 |
| CREATE INDEX　形式 2（インデクス定義）<br>CREATE INDEX　形式 3（部分構造インデクス定義） | ログレスモードで実行してください。<br>なお，左記の SQL 実行時は，特に大量のシステムログを出力します。そのため，システムログファイルの容量の過不足に関係なく，ログレスモードで運用することをお勧めします。ログレスモードで UAP 又はユティリティを実行するときの運用については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。 |
| DROP SCHEMA（スキーマ削除） | 指定した認可識別子のスキーマ内の次に示す定義を個別に削除し，一度に出力するシステムログ量を減らしてください。<br>• 実表<br>• ビュー表（パブリックビューも含む）<br>• 外部表<br>• インデクス<br>• 外部インデクス<br>• アクセス権限<br>• ルーチン（DROP SCHEMA で指定した認可識別子が定義したパブリック手続き，パブリック関数を含む）<br>• トリガ<br>• 抽象データ型<br>• インデクス型 |

## 17.1.2 表定義時に出力されるシステムログ量

表定義時に出力されるシステムログ量は，次に示す計算式で求めます。

### (1) HiRDB/シングルサーバの場合

| 条件 | システムログ量（単位：バイト） |
|---|---|
| 表を横分割しない場合 | ((1256 × b + 2500) × 1.2) × a + (632 × a × d) ※ |
| 表を横分割する場合 | ((1256 × b + 1800 × c + 2500) × 1.2) × a + (632 × a × d) ※ |

a：定義する表の総数（個）

b：定義する表の列数の平均値（個）

c：定義する表の平均分割数（個）

d：LOB 列を格納する RD エリア数（個）

注※　LOB 列を定義する表がある場合に加算します。

### (2) HiRDB/パラレルサーバの場合

| 条件 | システムログ量（単位：バイト） |
|---|---|
| ディクショナリサーバで出力されるシステムログ量 | ((1256 × b + 1800 × c + 2500) × 1.2) × a |
| バックエンドサーバで出力されるシステムログ量 | 912 × a × d + (632 × a × e) ※ |

a：定義する表の総数（個）

b：定義する表の列数の平均値（個）

c：定義する表の平均分割数（個）

d：定義する表のバックエンドサーバ内での分割数（個）

e：このバックエンドサーバ内にある RD エリアで，LOB 列を格納する RD エリア数（個）

注※　このバックエンドサーバ内に定義する表で，LOB 列を定義する表がある場合に加算します。

## 17.1.3　インデクス定義時に出力されるシステムログ量

インデクス定義時に出力されるシステムログ量は，次に示す計算式で求めます。

### (1) HiRDB/シングルサーバの場合

| 条件 | システムログ量（単位：バイト） |
|---|---|
| インデクスを横分割しない場合 | $\sum_{i=1}^{n} \{ 5624 + 800 \times b + 1256 \times \uparrow b \times 5 \div 24 \uparrow + 1256 \}$ ※<br>＋(132 + $X_i$ ×↑(100 − f) ÷ 100 ↑) × $W_i$ + 272 × $W_i$ + 1940 ×↑ $W_i$ ÷ $V_i$ ↑<br>} |

| 条件 | システムログ量（単位：バイト） |
|---|---|
| インデクスを横分割する場合 | $\sum_{i=1}^{n}\{5124+800\times b+500\times c+800\times a+1256\times\uparrow b\times 5\div 24\uparrow +1256^{※}+\sum_{j=1}^{a}\{(132+X_{ij})\times\uparrow(100-f)\div 100\uparrow)\times W_{ij}+272\times W_{ij}+1940\uparrow W_{ij}\div V_{ij}\uparrow\}\}$ |

注※　除外値指定がない場合は 0 になります。

a：インデクスを定義する表の横分割数（個）

b：インデクスの構成列数（個）

c：インデクスを格納する RD エリア数（個）

f：インデクス定義時に指定する PCTFREE オペランドの値（%）
　ページ内未使用領域比率の値です。

n：定義するインデクスの総数（個）

V：インデクスを格納するユーザ用 RD エリアのセグメントサイズ（ページ）

W：インデクスの格納ページ数（ページ）
　「16.1.3　インデクスの格納ページ数の計算方法」を参照してください。
　なお，CREATE INDEX で EMPTY を指定した場合は 0 になります。

X：インデクスを格納するユーザ用 RD エリアのページ長（バイト）

### （2） HiRDB/ パラレルサーバの場合

| 条件 | システムログ量（単位：バイト） |
|---|---|
| ディクショナリサーバで出力されるシステムログ量 | $\sum_{i=1}^{n}\{5124+800\times b+500\times c+800\times a+(1256\times\uparrow b\times 5\div 24\uparrow)+1256^{※}\}$ |
| バックエンドサーバで出力されるシステムログ量 | $\sum_{i=1}^{n}\{814+(132+X_i\times\uparrow(100-f)\div 100\uparrow)\times W_i+272\times W_i+1940\times\uparrow W_i\div V_i\uparrow\}$ |

注※　除外値指定がない場合は 0 になります。

a：インデクスを定義する表の横分割数（個）

b：インデクスの構成列数（個）

c：インデクスを格納する RD エリア数（個）

f：インデクス定義時に指定する PCTFREE オペランドの値（%）

ページ内未使用領域比率の値です。

n：定義するインデクスの総数（個）

V：インデクスを格納するユーザ用 RD エリアのセグメントサイズ（ページ）

W：インデクスの格納ページ数（ページ）
「16.1.3　インデクスの格納ページ数の計算方法」を参照してください。
なお，CREATE INDEX で EMPTY を指定した場合は 0 になります。

X：インデクスを格納するユーザ用 RD エリアのページ長（バイト）

## 17.1.4　表データ更新時に出力されるシステムログ量

表中の行を操作すると次の表に示すシステムログが出力されます。

**表 17-1　表中の行を操作したときに出力されるシステムログの種類**

| システムログの種類 | 説　明 |
|---|---|
| 基本行ログ | 基本行ログは，表中の行データを追加，削除又は更新するときに出力されます。 |
| 分岐行ログ | 分岐行ログは，次に示すデータ型のデータを操作するときに出力されます。<br>• VARCHAR※1<br>• NVARCHAR※2<br>• MVARCHAR※1<br>• 繰返し列<br>• 抽象データ型<br>• BINARY 型※3 |
| インデクスログ | インデクスログは，インデクスのキーを追加，削除又は更新するときに出力されます。インデクスログ量は，データベースの操作内容（INSERT 文，DELETE 文及び UPDATE 文）によって，表 17-2 に示すとおりに求めてください。 |
| イベントログ | イベントログは，HiRDB Datareplicator を使用しているとき，繰返し列を含む行データを追加，削除又は更新すると出力されます。 |

注※ 1　次に示すどちらかの条件を満たす場合に分岐行ログが出力されます。
- ノースプリットオプションの指定がなく，実際のデータ長が 256 バイト以上の場合
- ノースプリットオプションの指定があり，実際の 1 行のデータ長の合計がページ長を超える場合

注※ 2　次に示すどちらかの条件を満たす場合に分岐行ログが出力されます。
- ノースプリットオプションの指定がなく，実際のデータ長が 128 文字以上の場合
- ノースプリットオプションの指定があり，実際の 1 行のデータ長の合計がページ長を超える場合

注※ 3　実際の 1 行のデータ長の合計がページ長を超える場合に分岐行ログが出力されます。

**表 17-2　データベースの操作内容と求めるログ量**

| 操作内容（SQL 文） | 求めるログ量 |
|---|---|
| キーの追加（INSERT 文） | 追加ログ量 |
| キーの削除（DELETE 文） | 削除ログ量 |
| キーの更新（UPDATE 文） | 更新前キーの削除ログ量＋更新後キーの追加ログ量 |

データベースを更新（INSERT，DELETE 及び UPDATE）する場合に出力されるシステムログ量は，操

作内容（INSERT，DELETE 及び UPDATE）によって異なります。システムログ量の計算式を次に示します。

なお，ログを取得しない UAP 実行時に出力されるシステムログ量は，セグメント確保が発生する場合のログ量として 460 バイトとなります。

| 条件 | 計算式（単位：バイト） |
|---|---|
| 表に n 行追加（INSERT）する場合又は表から n 行削除（DELETE）する場合に出力されるシステムログ量 | （a ＋ b ＋ c）× n |
| 表中の n 行を更新（UPDATE）する場合に出力されるシステムログ量 | （a※1 ＋ d※2 ＋ e※3）× n |

a：基本行ログ量（バイト）

b：全分岐行ログ量の合計値（バイト）

c：全インデクスログ量の合計値（バイト）

d：更新対象分岐行ログ量の合計値（バイト）

e：更新対象インデクスログ量の合計値（バイト）

n：操作する行数（件）

注※ 1　更新（UPDATE）する列値が基本行内にある場合に加算します。

注※ 2　更新（UPDATE）する列値が分岐行内にある場合に加算します。

注※ 3　更新（UPDATE）する列にインデクスを定義する場合に加算します。

## （1）基本行ログ量の見積もり

データ 1 件当たりの基本行ログ量の計算式を次の表に示します。

**表 17-3　データ 1 件当たりの基本行ログ量の計算式**

| データの操作内容（SQL 文） | | 出力されるログ量（単位：バイト） |
|---|---|---|
| データの追加（INSERT 文） | | k ＋ 152 |
| データの削除（DELETE 文） | | |
| データの更新（UPDATE 文） | FIX 表で f ≦ 12 | $\sum_{i=1}^{f} d_i + \sum_{j=1}^{f} d_j + 4 \times f + 152$ |
| | 非 FIX 表 又は f ＞ 12 | k1 ＋ k2 ＋ 160<br>● pd_nowait_scan_option オペランドに LOCK を指定する場合に加算します。<br>314 × 2 |
| | 行インタフェース使用時 | 2 × k1 ＋ 160 |

k：追加又は削除する行長

k1：更新する行の更新前行長

k2：更新する行の更新後行長

f：更新する列数

$d_i$：更新する列の更新前データ長

$d_j$：更新する列の更新後データ長

注 1

k，k1，k2 の値は，操作する表に FIX 指定をするかどうかによって異なります。それぞれの計算式を次に示します。なお，表のデータ長については次を参照してください。

- 「表 16-1　データ長一覧」
- 「表 16-2　可変長文字列型のデータ長一覧（抽象データ型及び繰返し列を除く）」
- 「表 16-3　可変長文字列型のデータ長一覧（抽象データ型の場合）」
- 「表 16-4　可変長文字列型のデータ長一覧（繰返し列の場合）」
- **FIX 指定をする場合**
  表の全列のデータ長の合計値＋ 4
- **FIX 指定をしない場合**
  表の全列のデータ長の合計値＋ 6 ＋ 2 ×表の全列数

注 2

HiRDB Datareplicator を使用している場合，FIX 表に対して 12 列以下の UPDATE を列単位で実行すると，13 列以上の UPDATE と同量のログが出力されます。

注 3

表に BLOB 列が定義されている場合，表 17-3 の行長は BLOB 列のとき 9 バイト固定とし，次の表に示すログ量を加算してください。

**表 17-4　BLOB 列のデータ 1 件当たりのログ量の計算式**

| データの操作内容（SQL 文） | recovery の指定 | 出力されるログ量（単位：バイト） |
|---|---|---|
| データの追加（INSERT 文） | なし又は partial | 604 ＋ 180 × p5 |
| | all | 2348 ＋ p1 ＋ 8340 × p2 ＋（148 ＋ lt）× p3 |
| | no | 300 |
| データの削除（DELETE 文） | なし又は partial | 460 ＋ 180 × p6 |
| | all | |
| | no | 468 |
| データの更新（UPDATE 文） | なし又は partial | 1324 |
| | all | 604 ＋ 180 × p5 ＋ 460 ＋ 180 × p6 |
| | no | 312 |
| データの連結演算（UPDATE 文）Bb ≦ 7168 | なし又は partial | 2344 |
| | all | 8340 × a ＋ 1600 ＋ d ＋ 8340 × p4 ＋（148 ＋ lt2）× p3 |
| | no | 428 |
| データの連結演算（UPDATE 文）Bb ＞ 7168 | なし又は partial | 2772 |
| | all | 2772 ＋ Ba ＋ 8340 × p4 ＋（148 ＋ lt2）× p3 |
| | no | 428 |

| データの操作内容（SQL 文） | recovery の指定 | 出力されるログ量（単位：バイト） |
|---|---|---|
| データの後方削除更新（UPDATE 文）<br>Bb ≦ 7168 | なし又は partial | 2344 |
| | all | 9512 |
| | no | 428 |
| データの後方削除更新（UPDATE 文）<br>Bb ＞ 7168 | なし又は partial | 2492 ＋ 180 ×↑（Bb-Bd）÷ 8192 ↑ |
| | all | 2492 ＋ 180 ×↑（Bb-Bd）÷ 8192 ↑ |
| | no | 428 ＋ 180 ×↑（Bb-Bd）÷ 8192 ↑ |

Bi：BLOB データ長（単位：バイト）

Ba：次のどちらかの値

- Bb ＞ 7168 の場合：8192 － {(Bb － 7168）－↓（Bb － 7168）÷ 8192 ↓× 8192}
- Bb ≦ 7168 の場合：0

Bb：BLOB 更新前データ長（単位：バイト）

Bc：BLOB 追加データ長（単位：バイト）

Bd：更新後データ長（SUBSTR 関数の値式 3 の指定値）（単位：バイト）

lt：次のどちらかの値

- Bi ＞ 7168 の場合：Bi － 7168 －↓（Bi － 7168）÷ 8192 ↓× 8192
- Bi ≦ 7168 の場合：0

lt2：次のどちらかの値

- Bc ＋ Bb ＞ 7168 の場合：(Bc ＋ Bb － Ba － 7168）－↓（Bc ＋ Bb － Ba － 7168）÷ 8192 ↓× 8192
- Bc ＋ Bb ≦ 7168 の場合：0

p1：次のどちらかの値

- Bi ＞ 7168 の場合：7168
- Bi ≦ 7168 の場合：Bi

p2：次のどちらかの値

- Bi ＞ 7168 の場合：↓（Bi － 7168）÷ 8192 ↓
- Bi ≦ 7168 の場合：0

p3：次のどちらかの値

- lt ＝ 0 又は lt2 ＝ 0 の場合：0
- lt ＞ 0 又は lt2 ＞ 0 の場合：1

p4：次のどちらかの値

- Bc ＋ Bb ＞ 7168 の場合：↓（Bc ＋ Bb － Ba － 7168）÷ 8192 ↓
- Bc ＋ Bb ≦ 7168 の場合：0

p5：次のどちらかの値

- Bi ＞ 0 の場合：↑（Bi ＋ 1024）÷ 8192 ↑
- Bi ＝ 0 の場合：0

p6：次のどちらかの値

- Bb ＞ 0 の場合：↑（Bb ＋ 1024）÷ 8192 ↑
- Bb ＝ 0 の場合：0

a：次のどちらかの値
- Bb ＞ 0 の場合：1
- Bb ＝ 0 の場合：0

d：次のどちらかの値
- Bc ＋ Bb ≦ 7168 の場合：Bc ＋ Bb
- Bc ＋ Bb ＞ 7168 の場合：7168

## (2) 分岐行ログ量の見積もり

### (a) ノースプリットオプション指定なしの VARCHAR，NVARCHAR 及び MVARCHAR の分岐行ログ量

発生する分岐行分のログ量を算出してその合計値を求めます。1 分岐行ログ量の計算式を次の表に示します。

**表 17-5　1 分岐行のログ量の計算式（その 1）**

| データの操作内容（SQL 文） | | 出力されるログ量（単位：バイト） |
|---|---|---|
| データの追加（INSERT 文） | | k ＋ 152 |
| データの削除（DELETE 文） | | |
| データの更新（UPDATE 文） | 更新によって新たな分岐行ができる場合 | k2 ＋ 160 |
| | 分岐行を更新する場合 | k1 ＋ k2 ＋ 160 |
| | 更新によって分岐行が削除される場合 | k1 ＋ 160 |

k：追加又は削除する一つの分岐行長

k1：更新する一つの更新前分岐行長

k2：更新する一つの更新後分岐行長

注　k，k1 及び k2 の行長は，次に示す計算式で求めます。
8 ＋ MIN（平均データ長の実長，RD エリアのページ長− 48 ）

### (b) 抽象データ型の列，繰返し列，BINARY 型の列，ノースプリットオプション指定ありの VARCHAR，NVARCHAR 及び MVARCHAR の分岐行ログ量

発生する分岐行分のログ量を算出してその合計値を求めます。1 分岐行ログ量の計算式を次の表に示します。

ただし，表格納 RD エリアが複数あり，ページ長が各 RD エリアで異なる場合は，同じページ長の RD エリアごとに計算して，その合計を分岐行ログ量として求めてください。

**表 17-6　1 分岐行のログ量の計算式（その 2）**

| データの操作内容（SQL 文） | 出力されるログ量（単位：バイト） |
|---|---|
| データの追加（INSERT 文） | SPN ×（b ＋ 152） |
| データの削除（DELETE 文） | |
| データの更新（UPDATE 文） | SPN ×（b ＋ 160） |
| データの連結演算（UPDATE 文）※ | （b ＋ 160）＋（SPN − 1）×（b ＋ 152）<br>● pd_rpl_hdepath オペランドを指定する場合に加算します。<br>160 |

| データの操作内容（SQL 文） | 出力されるログ量（単位：バイト） |
|---|---|
| データの後方削除更新（UPDATE 文）※ | （b × 2 ＋ 160）＋（↓（c － d）÷（b － 57）↓）×（b ＋ 152）<br>● pd_rpl_hdepath オペランドを指定する場合に加算します。<br>160 |

注※　BINARY 型の列の場合だけです。

b：RD エリアのページ長

c：更新前データ長

d：更新後データ長（SUBSTR 関数の値式 3 の指定値）

SPN：求め方を次に示します。

分岐する列がある場合（分岐する条件については「表 16-1　データ長一覧」の注※ 5 で説明），INSERT 文及び DELETE 文は表を構成するすべての列について，UPDATE 文は更新対象列について次に示す値を求めてください。ただし，BINARY 型の列の連結演算の場合，di は追加するデータ長として計算してください。

SPN ＝ SPN1 ＋ SPN2

SPN1，及び SPN2 については「16.1.2　表の格納ページ数の計算方法」の「(3) 計算式中で使用する変数」を参照してください。ただし，a の値には 1 を設定してください。

## (3) インデクスログ量の見積もり

1 本のインデクスの 1 行ごとの操作で出力するインデクスログ量は，次の表に示す計算式で求めます。

**表 17-7　1 本のインデクスログ量の計算式**

| キーの操作内容（SQL 文） | | | 出力されるログ量（単位：バイト） |
|---|---|---|---|
| キーの追加（INSERT 文） | 新しいキーを追加する場合 | | k1 ＋ 156<br>又は（k1 ＋ 156）× 2 ※ |
| | 追加するキーと同じ行が既にある場合 | d ≦ 200 | k1 ＋ 156 |
| | | d ＞ 200 | k1 ＋ 292 |
| キーの削除（DELETE 文） | キー値を削除する場合 | | k2 ＋ 156 |
| | 削除するキー値と同じ値を持つ行がほかにある場合 | d ≦ 200 | k2 ＋ 156 |
| | | d ＞ 200 | k2 ＋ 292 |
| キーの更新（UPDATE 文） | | | キー削除のログ量＋キー追加のログ量 |

d：キー値の重複数

k1：追加するキー長（バイト）

k2：削除するキー長（バイト）

注

ここでいうキー長とは，DB 格納キー長のことです。キー長の求め方については，「16.1.3　インデク

スの格納ページ数の計算方法」を参照してください。

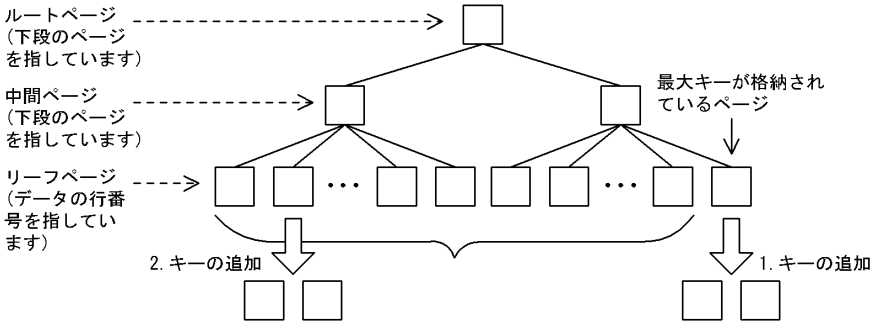
注※

インデクスキー無排他を適用した UNIQUE 指定のインデクスのときの計算式です。

### (a) インデクスページスプリット時のインデクスログ量の見積もり

**インデクスページスプリット**の概念を次の図に示します。

**図 17-1　インデクスページスプリットの概念**




〔説明〕

1. リーフページの一番右側（図中の最大キーが格納されているページ）にキーが追加され，ページが二つに分割されていることを最大キーが入っているページスプリットといいます。
2. そのほかのリーフページにキーが追加され，ページが二つに分割されていることを最大キーが入っていないページスプリットといいます。

インデクス格納ページをスプリットする場合，HiRDB は次に示す二つの方法でキー値を格納します。

**●最大キー値が入っていないページスプリットの場合**

キー値を追加又は削除すると，キーと未使用領域の比率をおよそ 50：50 にスプリットしてキー値を格納します。最大キー値が入っていないページスプリットが発生する条件を次に示します。

- インデクス格納ページ内にキーが入らないとき
- 重複するキー値が 201 件以上あり，同じキー値を持つ行を a 件以上追加するとき

a は次に示す計算式で求めます。このとき，a 件に 1 回はスプリットが発生します。

**計算式**

a ＝ ↑**インデクスを格納する** RD **エリアのページ長（バイト）** ÷ 4 ↑

**●最大キー値が入っているページスプリットの場合**

最大キー値が格納されているインデクス格納ページに対してキー値を追加又は更新すると，キーと未使用領域の比率を CREATE INDEX の PCTFREE オペランドの指定値に従ってスプリットし，キー値を格納します。

例えば，PCTFREE ＝ 30 と指定すると，キーと未使用領域の比率をおよそ 70：30 にスプリットしてキー値を格納します。

キーの追加によって CREATE INDEX の PCTFREE オペランドに指定する分の空き領域が確保できないときに，最大キー値が入っているページスプリットが発生します。ただし，上位レベルページは該当しません。

スプリット種別による 1 回当たりのインデクスログ量の計算式を次の表に示します。

**表 17-8　スプリット種別による 1 回当たりのインデクスログ量の計算式**

| スプリット種別 | 条件 | | | 出力されるログ量（単位：バイト） |
|---|---|---|---|---|
| 最大キーが入っているページのスプリット | インデクス中のキー値と異なるキーを追加した場合 | 上位ページにキーを追加するための空き領域が未使用領域にある場合 | 2 × k1 ＋ a ＋ 8 ×（m ＋ 1）≦ 31516 の場合 | 2 × k1 ＋ 472 ＋ a ＋ 8 ×（m ＋ 1） |
| | | | 2 × k1 ＋ a ＋ 8 ×（m ＋ 1）＞ 31516 の場合 | 2 × k1 ＋ 632 ＋ a ＋ 8 ×（m ＋ 1） |
| | | 上位ページにキーを追加するための空き領域が未使用領域にない場合 | 2 × k1 ＋ a ＋ 8 ×（m ＋ 1）≦ 31516 の場合 | $\sum_{i=2}^{n-1}$（288 ＋ a）＋ 2 × k1 ＋ 472 ＋ a ＋ 8 ×（m ＋ 1） |
| | | | 2 × k1 ＋ a ＋ 8 ×（m ＋ 1）＞ 31516 の場合 | $\sum_{i=2}^{n-1}$（288 ＋ a）＋ 2 × k1 ＋ 628 ＋ a ＋ 8 ×（m ＋ 1） |
| | インデクス中に追加するキー値と同じキーがある場合 | d1 ≦ 200 | 上位ページにキーを追加するための空き領域が未使用領域にある場合 | 2 × k1 ＋ 472 ＋ a ＋ 8 ×（m ＋ 1） |
| | | | 上位ページにキーを追加するための空き領域が未使用領域にない場合 | $\sum_{i=2}^{n-1}$（288 ＋ a）＋ 2 × k1 ＋ 472 ＋ a ＋ 8 ×（m ＋ 1） |
| | | d1 ＞ 200 | 下位ページにキーを追加するための空き領域が未使用領域にある場合 | k1 ＋ 472 ＋ a |
| | | | 下位ページにキーを追加するための空き領域が未使用領域にない場合 | k1 ＋ 462 ＋ 2 × a |
| 最大キーが入っていないページのスプリット | キーを追加するための空き領域が未使用領域にない場合 | 上位ページにキーを追加するための空き領域が未使用領域にある場合 | 2 × k1 ＋ a ＋ 8 ×（m ＋ 1）≦ 31516 の場合 | 2 × k1 ＋ 332 ＋ a ＋ 8 ×（m ＋ 1） |
| | | | 2 × k1 ＋ a ＋ 8 ×（m ＋ 1）＞ 31516 の場合 | 2 × k1 ＋ 492 ＋ a ＋ 8 ×（m ＋ 1） |
| | | 上位ページにキーを追加するための空き領域が未使用領域にない場合 | 2 × k1 ＋ a ＋ 8 ×（m ＋ 1）≦ 31516 の場合 | $\sum_{i=2}^{n-1}$（288 ＋ a）＋ 2 × k1 ＋ 332 ＋ a ＋ 8 ×（m ＋ 1） |
| | | | 2 × k1 ＋ a ＋ 8 ×（m ＋ 1）＞ 31516 の場合 | $\sum_{i=2}^{n-1}$（288 ＋ a）＋ 2 × k1 ＋ 492 ＋ a ＋ 8 ×（m ＋ 1） |

ａ：インデクスを格納している RD エリアのページ長（バイト）

d1：キー値の重複数

k1：追加するキー長（バイト）

ここでいうキー長とは，DB 格納キー長のことです。キー長の求め方については，「16.1.3　インデクスの格納ページ数の計算方法」を参照してください。

m：スプリットが発生した時点のインデクス段数

n：スプリットが波及した上位ページの段数
リーフページのスプリットによって波及した上位ページが，さらにスプリットした場合はn＝3（n≧3）となります。

注
この計算式は，1件当たりの行及びインデクス部分の更新ログ量の見積もり式です。このため，この計算式には，追加及び更新によって，新しいページ又はセグメントを確保したときのシステム管理情報に関するログ量を含んでいません。したがって，大量のデータを扱う場合は，次の表に示すログ量を加算して見積もる必要があります。

**表 17-9　ページ／セグメントの確保ログ量の計算式**

| 条件 | 出力されるログ量（単位：バイト） |
|---|---|
| データの追加（INSERT）又は更新（UPDATE）によって，行の格納ページを新しく確保する場合 | 440 |
| データの追加（INSERT）又は更新（UPDATE）によって，インデクスのページスプリットが発生する場合 | 544 × n ＋ 272 |
| 上記のページ確保に対して，セグメントの確保が発生する場合（セグメントサイズ数分のページ確保をするごとに発生します） | 1940 |

n：ページスプリットが発生した時点のインデクスの段数＋1

## (4) イベントログ量の見積もり

イベントログはHiRDB Datareplicatorを使用しているときに，繰返し列を含む行データを追加，削除又は更新すると出力されます。1行を操作したときに出力されるイベントログ量を次の表に示します。

**表 17-10　1行を操作したときに出力されるイベントログ量**

| データ操作内容 | | イベントログ量（単位：バイト） |
|---|---|---|
| データの追加（INSERT文） | | 156 × n |
| データの更新（UPDATE文） | 更新によって新しい要素が追加される場合（UPDATE ADD） | 164 × n |
| | 更新によって要素が削除される場合（UPDATE DELETE） | $\sum_{i=1}^{n}$（p5 +160） |
| | 指定した要素だけを更新する場合（UPDATE SET） | $\sum_{i=1}^{n}$（p5 +160） |
| | 指定した繰返し列を更新する場合（UPDATE SET） | 164 × n |

n：更新対象となる繰返し列の数

p5：↑｛↑（指定する最大添字番号の平均値－指定する最小添字番号の平均値＋1）÷8↑｝÷4↑×4

### 17.1.5 ユティリティによるデータベース作成時に出力されるシステムログ量

次に示すユティリティを実行する場合，表 17-11 に示すシステムログが出力されます。

- データベース作成ユティリティ（pdload コマンド）
- データベース再編成ユティリティ（pdrorg コマンド）
- リバランスユティリティ（pdrbal コマンド）

システムログ量は表 17-11 で求めた値に，表 17-12 に示す値を加算して算出します。なお，表及びインデクスを横分割している場合は，表及びインデクスを格納する RD エリアごとにシステムログ量を計算します。

- HiRDB/ シングルサーバの場合
  RD エリアごとに計算したシステムログ量の合計を求めます。
- HiRDB/ パラレルサーバの場合
  RD エリアごとに計算したシステムログ量の合計をサーバ単位に求めます。

**表 17-11 ユティリティによるデータベース作成時に出力されるシステムログ量の計算式**

| 項番 | 条件 | 出力されるシステムログ量（単位：バイト） | |
|---|---|---|---|
| | | -l オプションに a を指定 | -l オプションに p を指定 |
| 1 | インデクスを一括作成する場合（-i オプションに c 又は n を指定） | $\sum_{i=1}^{n} \{ [132 + \uparrow (100 - f) \div 100 \uparrow \times X_i] \times W_i + 280 \times W_i + 1940 \times \uparrow W_i \div V_i \uparrow \} + 280 \times m + 1940 \times \uparrow m \div s \uparrow + a \times r$ | $\sum_{i=1}^{n} \{ 280 \times W_i + 1940 \times \uparrow W_i \div V_i \uparrow \} + 280 \times m + 1940 \times \uparrow m \div s \uparrow + c \times r$ |
| 2 | インデクスを一括作成しない場合（-i オプションに s を指定） | $\sum_{i=1}^{n} \{ 280 \times W_i + 1940 \times \uparrow W_i \div V_i \uparrow + b \times r + e \times d \} + 280 \times m + 1940 \times \uparrow m \div s \uparrow + a \times r$ | $\sum_{i=1}^{n} \{ 280 \times W_i + 1940 \times \uparrow W_i \div V_i \uparrow + b \times r + e \times d \} + 280 \times m + 1940 \times \uparrow m \div s \uparrow + c \times r$ |
| 3 | インデクスを再作成，又は再編成する場合（pdload 0 件ロード，又は pdrorg -k オプションに ixrc 若しくは ixor を指定） | 次のシステムログ量を加算してください。<br>• 17.1.3 インデクス定義時に出力されるシステムログ量<br>• 表 17-9 ページ／セグメントの確保ログ量の計算式<br>• 表 17-14 PURGE TABLE 文実行時に出力されるシステムログ量 のインデクスのシステムログ量 | 次のシステムログ量を加算してください。<br>• 表 17-9 ページ／セグメントの確保ログ量の計算式<br>• 表 17-14 PURGE TABLE 文実行時に出力されるシステムログ量 のインデクスのシステムログ量 |

注

インデクスの一括作成又はインデクスの作成時のシステムログ量は，作成するインデクス数分の計算が必要です。

ａ：「表 17-3　データ 1 件当たりの基本行ログ量の計算式」，「表 17-4　BLOB 列のデータ 1 件当たりのログ量の計算式」で求めた，データを 1 件追加するときに出力されるログ量

ｂ：「表 17-7　1 本のインデクスログ量の計算式」で求めたデータを 1 件追加するときに出力されるログ量

ｃ：「表 17-4　BLOB 列のデータ 1 件当たりのログ量の計算式」で求めた，データを 1 件追加するときに出力されるログ量

ｄ：インデクススプリット時に出力されるシステムログ量
「表 17-8　スプリット種別による 1 回当たりのインデクスログ量の計算式」及び「表 17-9　ページ／セグメントの確保ログ量の計算式」を参照してください。

ｅ：インデクスのスプリット回数

ｆ：インデクス定義時に指定する PCTFREE オペランド（ページ内未使用領域比率）の値（%）

ｍ：表の格納ページ数（ページ）
「16.1.2　表の格納ページ数の計算方法」を参照してください。

ｎ：表に定義されているインデクス数（個）

ｒ：表に格納する行数（行）

ｓ：表を格納するユーザ用 RD エリアのセグメントサイズ（ページ）

$V_i$：インデクスを格納するユーザ用 RD エリアのセグメントサイズ（ページ）

$W_i$：インデクスの格納ページ数（ページ）
「16.1.3　インデクスの格納ページ数の計算方法」を参照してください。

$X_i$：インデクスを格納するユーザ用 RD エリアのページ長（バイト）

表 17-12 の項番 1，及び項番 2 の場合に，システムログ量の算出時に加算する値と，加算する場合の条件を次の表に示します。

**表 17-12　システムログ量の算出時に加算する値と加算する場合の条件**

| 加算する場合の条件 | システムログ量に加算する値 |
|---|---|
| LOB 列を定義している表の場合 | 次に示すログ量を（LOB 列数×行数）の数分加算してください。<br>•「表 17-4　BLOB 列のデータ 1 件当たりのログ量の計算式」の「データの追加（INSERT 文）」<br>• LOB 列の回復属性が recovery no 以外の場合<br>3544 ×（LOB データ長÷ 31744）（単位：バイト） |
| データベース作成ユティリティ（pdload）を -d オプション指定で，又はデータベース再編成ユティリティ（pdrorg）を実行する場合 | 「表 17-14　PURGE TABLE 文実行時に出力されるシステムログ量」の「表」又は「リバランス表」と，対象表に定義されている全インデクス，及び全 LOB 列（又は LOB 属性）のログ量を加算してください。 |
| データベース再編成ユティリティ（pdrorg）で LOB 列，又は LOB 属性がある表を再編成する場合で次の二つの条件を満たすとき | 次に示すログ量を LOB 用 RD エリアの構成ファイル数分加算してください。<br>Σ｛17000 ×↑（HiRDB ファイルのセグメント数÷ 64000）↑× 95｝ |

| 加算する場合の条件 | システムログ量に加算する値 |
|---|---|
| • -j オプションを指定しないで再編成する<br>• LOB 列が recovery no 指定でなく，かつ -l オプションが n 指定でない | 次に示すログ量を（LOB 列又は LOB 属性の数×行数）の数分加算してください。<br>● LOB 列が recovery all 指定かつ -l a 指定の場合<br>↑ 17600 ÷ sr ↑× sr<br>●上記以外の場合<br>↑ 3200 ÷ sr ↑× sr<br>sr：pd_log_rec_leng オペランドで指定するシステムログファイルのレコード長 |
| HiRDB Datareplicator 連携機能を使用（pd_rpl_hdepath オペランドを指定）していて，表に繰返し列を含む場合 | 「表 17-10　1 行を操作したときに出力されるイベントログ量」に示すログ量を追加行数分加算してください。 |

## 17.1.6　SQL 操作に応じて出力されるシステムログ量

システムログは，システム共通定義の pdhibegin オペランドに -k cnc を指定している状態で，CONNECT，DISCONNECT 又は set session authorization を実行した場合に出力されます。SQL 操作に応じて出力されるシステムログ量を求める計算式を次に示します。

**計算式**

システムログ量（単位：バイト）＝ 568 ×（CONNECT 回数＋ set session authorization 実行回数）

## 17.1.7　RD エリアの自動増分機能使用時に出力されるシステムログ量

RD エリアの自動増分機能を使用すると，自動増分の実行時にシステムログが出力されます。出力されるシステムログ量を求める計算式を次の表に示します。

**表 17-13　RD エリアの自動増分機能使用時に出力されるシステムログ量**

| RD エリアの種類 | システムログ量（単位：バイト） |
|---|---|
| LOB 用 RD エリアの場合 | 1956 |
| LOB 用 RD エリア以外の場合 | 1372 ＋（144 ＋ p）× 2 |

p：自動増分する RD エリアのページサイズ

## 17.1.8　PURGE TABLE 文実行時に出力されるシステムログ量

PURGE TABLE 文実行時に出力されるシステムログ量は，表に定義されているすべてのインデクス，LOB 列，又は LOB 属性分のログ量を合計して求めます。分割表，及び分割インデクスの場合は RD エリアごとに求めたログ量を合計して求めます。PURGE TABLE 文実行時のシステムログ量の計算式を次の表に示します。なお，計算式中の変数 A，B，C は RD エリア構成ファイルごとに求めてください。

**表 17-14　PURGE TABLE 文実行時に出力されるシステムログ量**

| 種別 | システムログ量（単位：バイト） |
|---|---|
| 表 | 1000 ＋ 1100 ×確保セグメント数 |
| リバランス表 | 1000 ＋ 1100 ×確保セグメント数＋ 400 ×↑ 1024 ÷分割 RD エリア数↑ |
| インデクス※ | 1000 ＋ 1100 ×確保セグメント数 |

| 種別 | システムログ量（単位：バイト） |
|---|---|
| LOB 列又は LOB 属性※ | 1000 ＋ 17000 × A ＋ 160 × B ＋ 2 × C |

A：↑（HiRDB ファイルの使用中セグメント数÷ 64000）↑× 95

B：↑（HiRDB ファイルの使用中セグメント数÷ 64000）↑

C：↑（HiRDB ファイルの使用中セグメント数÷ 64000）↑× 8150

注※

プラグインの場合は，各プラグインの初期化のログが出力されます。各プラグインのマニュアルを参照してください。

## 17.1.9　空きページ解放ユティリティ（pdreclaim）実行時に出力されるシステムログ量

空きページ解放ユティリティ（pdreclaim）による表，インデクスの空きページ，又はセグメントの解放をする場合，システムログを出力します。

指定したオプションと対象となる資源の組み合わせによるログ量を次に示します。なお，ログ量の説明の項番は，表 17-15 の項番に対応しています。

| オプション | 対象資源 | ログ量 |
|---|---|---|
| なし | 表 | 項番 (1) の値 |
| | インデクス | 項番 (2) の値 |
| -j | 表 | 項番 (3) の値 |
| | インデクス | |
| -a | 表 | 項番 (1) と (4) を合計した値 |
| | インデクス | 項番 (2) と (4) を合計した値 |

ログ量を求める計算式を次の表に示します。なお，表及びインデクスを横分割している場合は分割した RD エリアごとに計算する必要があります。HiRDB/ シングルサーバの場合は，横分割した RD エリアごとに計算したログ量の合計を求めます。HiRDB/ パラレルサーバの場合は，横分割した RD エリアごとに計算したログ量をサーバごとに求め，その合計を求めます。

**表 17-15　空きページ解放ユティリティ実行時に出力されるシステムログ量**

| 項番 | 種別 | | システムログ量（単位：バイト） |
|---|---|---|---|
| (1) | 表 | | 140 × n<br>● **-o オプションを指定していない場合に加算します。**<br>504 × m |
| (2) | インデクス | キー値の重複が 201 以上のキーが存在しない | {((k ＋ 14) ÷ (j － 68)) × 1408 ＋ k ＋ 12 × h ＋ 908} × n<br>● **-x オプションを指定している場合に加算します。**<br>＋ {m － n －（((k ＋ 14) × 2 × m) ÷ (j － 68))} ×（j ＋ 76）× 0.7 |
| | | キー値の重複が 201 以上のキーが存在する | (724 ＋ k) × n ＋ 285 × MAX（n，16 × n ÷ (j － 78)) － 159<br>● **-x オプションを指定している場合に加算します。**<br>＋ {m － n －（((k ＋ 14) × 2 × m) ÷ (j － 68))} ×（j ＋ 76）× 0.7 |

| 項番 | 種別 | | システムログ量（単位：バイト） |
|---|---|---|---|
| (3) | セグメント | -j オプション指定時 | 2380 × p |
| (4) | | -a オプション指定時 | 3100 × p |

h：インデクスの段数（段）

j：ページサイズ（バイト）

k：インデクスキーの長さ（バイト）

m：使用中ページ数（満杯ページを除く）（個）

n：使用中空きページ数（個）

p：使用中空きセグメント数（個）
解放途中セグメントの数を含みます。

## 17.1.10 再編成時期予測機能使用時に出力されるシステムログ量

再編成時期予測機能を使用する場合，HiRDB/ パラレルサーバの場合はディクショナリサーバで出力するログ量，HiRDB/ シングルサーバの場合はシングルサーバで出力するログ量に，次の計算式で求められるログ量を加算する必要があります。

**計算式**

```
システムログ量（単位：バイト）＝
  n × {1604 ×（A ＋ B ＋ C ＋ D）×（↑ E ↑＋ 1)}
  ＋ m × {872 ×（a ＋ b ＋ c ＋ 1)}
  ＋ 11680 ×↑ {(A ＋ B ＋ C ＋ D）×（↑ E ↑＋ 1)} × 30 ÷ 540 ↑
  ＋ 332 × {(A ＋ B ＋ C ＋ D）×（↑ E ↑＋ 1)} × 30
  ＋ 7760
```

A：作成した表数＋ 61

B：作成したインデクス数＋ 124

C：作成した表に定義した BLOB 列の総数＋ 3

D：作成した表に定義した BLOB 属性の総数

E：表を格納する RD エリアの分割数の平均値
分割していない場合は 1 とします。また，平均値は切り上げます。

a：SQL 又はコマンドが処理した表を格納している RD エリア数

b：SQL 又はコマンドが処理したインデクスを格納している RD エリア数

c：SQL 又はコマンドが処理した表を格納する LOB 用 RD エリア数

n：状態解析結果蓄積機能（pddbst -k logi -e）を実行した回数

m：運用履歴表を更新する SQL 又はコマンドの実行回数
運用履歴表を更新する SQL 及びコマンドについては，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

# 17.2　シンクポイントダンプファイルの容量の見積もり

## (1) シンクポイントダンプファイルの容量の求め方

シンクポイントダンプファイルの容量は，次に示す計算式で求めます。

**計算式**

シンクポイントダンプファイルの容量（単位：バイト）
＝シンクポイントダンプファイルのレコード数※1 × 4096※2

注※1

「(2) シンクポイントダンプファイルのレコード数の求め方」を参照してください。ここで求めた値をpdloginit コマンドの -n オプションに指定します。

注※2

シンクポイントダンプファイルのレコード長です。

## (2) シンクポイントダンプファイルのレコード数の求め方

| 条件 | | レコード数の計算式 |
|---|---|---|
| HiRDB/ シングルサーバの場合 | | ↑｛1520 ×（2 × a ＋ 7)｝÷ 4096 ↑<br>＋↑（96 ＋ 112 × c）÷ 4096 ↑＋ 3 |
| HiRDB/ パラレルサーバの場合 | フロントエンドサーバの場合 | ↑｛728 ×（2 × a ＋ 7）＋ 64 ×（2 × a ＋ 7）×（b × 4 ＋ f ＋ 2)｝÷ 4096 ↑＋ 4 |
| | バックエンドサーバの場合 | ↑｛1456 ×（2 × d ＋ 7）＋ 64 ×（2 × d ＋ 7）×（b × 4 ＋ f ＋ 2)｝÷ 4096 ↑<br>＋↑（96 ＋ 112 × c）÷ 4096 ↑＋ 3 |
| | ディクショナリサーバの場合 | ↑｛1456 ×（2 × e ＋ 7）＋ 64 ×（2 × e ＋ 7）×（b × 4 ＋ f ＋ 2)｝÷ 4096 ↑<br>＋↑（96 ＋ 112 × c）÷ 4096 ↑＋ 3 |

ａ：pd_max_users オペランドの値
　マルチフロントエンドサーバの場合は pd_max_users オペランドの値＋ 1

ｂ：バックエンドサーバ数

ｃ：pd_lck_until_disconnect_cnt オペランドの値

ｄ：pd_max_bes_process オペランドの値

ｅ：pd_max_dic_process オペランドの値

ｆ：フロントエンドサーバ数

# 17.3　ステータスファイルの容量の見積もり

## (1) ステータスファイルの容量の求め方

ステータスファイルの容量は，次に示す計算式で求めます。

**計算式**

ステータスファイルの容量（単位：バイト）＝ a × b

a：ステータスファイルのレコード数
「(2) ステータスファイルのレコード数の求め方」を参照してください。ここで求めた値を pdstsinit コマンドの -c オプションに指定します。

b：ステータスファイルのレコード長
pdstsinit コマンドの -l オプションに指定する値です。

## (2) ステータスファイルのレコード数の求め方

ステータスファイルのレコード数は，次に示す計算式で求めます。

**計算式**

レコード数＝↑｛↑ S ÷（レコード長− 40）↑＋↑ S ÷ 100 ↑＋ S ＋ 2｝× 1.2 ↑

注　S については，「(3)S の求め方」を参照してください。

## (3) S の求め方

### (a) HiRDB/ シングルサーバの場合

| 種別 | S の計算式 |
|---|---|
| ユニット用 | ↑（2056 ＋ 128 × d）÷ g ↑＋↑ 2512 ÷ g ↑＋↑ 40 ÷ g ↑<br>＋↑ 308 ÷ g ↑＋↑ 4000 ÷ g ↑＋↑ 43536 ÷ g ↑＋↑ 32 ÷ g ↑ |

| 種別 | S の計算式 |
|---|---|
| サーバ用 | ↑ 128 ÷ g ↑＋↑ 3280 ※1 ÷ g ↑＋↑（592 ※2＋j × b）÷ g ↑<br>＋↑ 8192 ÷ g ↑＋↑（8192 − 128）÷ g ↑ × {↑（a ＋ m）÷ k ↑− 1}<br>＋↑［↑ {(↑ c ÷ 127 ↑＋ 1）× 2048} ÷ 8192 ↑× 8192］÷ g ↑<br>＋↑ 2048 ÷ g ↑＋↑ {244 ＋ 64 ×（2 × h ＋ 7)} ÷ g ↑<br>＋↑ 96 ÷ g ↑＋↑ {48 ＋ 68 ×（2 × h ＋ 7)} ÷ g ↑<br>＋↑ {12288 ＋↑（e × 98 ÷ 12288）↑× 12288<br>＋↑（f × 80 ÷ 12288）↑× 12288} ÷ g ↑<br>＋↑（8 × c ＋ 4）÷ g ↑＋↑ 256 ÷ g ↑<br>＋ w<br>**●ステータスファイルのレコード長＜** 4096 **の場合に加算します。**<br>＋↓ MAX {3400 ÷ r ＋ 0.7，1} ↓× MAX（↓ 4096 ÷ s ↓，2）× n<br>＋↓ 5662310 ÷ r ＋ 0.7 ↓× MAX（↓ 4096 ÷ s ↓，2）× p<br>**●** 4096 **≦ステータスファイルのレコード長＜** 12288 **の場合に加算します。**<br>＋↓ MAX {(3400 ÷↓（s − 348）÷ 20 ↓）＋ 0.7，1} ↓× n<br>＋↓（5662310 ÷↓（s − 348）÷ 20 ↓）＋ 0.7 ↓× p<br>**●** 12288 **≦ステータスファイルのレコード長の場合に加算します。**<br>＋↓ MAX {(3400 ÷↓（s − 876）÷ 20 ↓）＋ 0.7，1} ↓× n<br>＋↓（5662310 ÷↓（s − 876）÷ 20 ↓）＋ 0.7 ↓× p<br>**●** pd_log_auto_unload_path **オペランドを指定する場合に加算します。**<br>＋↑ 20848 ÷ g ↑<br>**●高速系切り替え機能及びスタンバイレス型系切り替え機能を使用する場合に加算します。**<br>＋↑（256 × i ＋ 4096）÷ g ↑<br>**●** pd_dbbuff_modify **オペランドに** Y **を指定する場合に加算します。**<br>＋ v |

a：pdlogadfg オペランドの指定数

b：pdlogadfg -d spd オペランドの指定数

c：pd_max_rdarea_no オペランドの値

d：90

e：pdbuffer オペランドの指定数

f：pdbuffer オペランドの -i オプション指定数

g：ステータスファイルのレコード長− 40

h：pd_max_users の値

i：pd_max_rdarea_no オペランドの値

j：736

k：11

m：1

n：サーバ内の RD エリア数≦ 3400 の場合は 1
3401 ≦サーバ内の RD エリア数≦ 6800 の場合は 2
6801 ≦サーバ内の RD エリア数の場合は 3

p：サーバ内の RD エリア数≦ 10200 の場合は 0
10201 ≦サーバ内の RD エリア数≦ 5672510 の場合は 1

5672511 ≦サーバ内の RD エリア数≦ 11334820 の場合は 2
11334821 ≦サーバ内の RD エリア数の場合は 3

r：↓（s － 348）÷ 20 ↓＋↓ g ÷ 20 ↓×（MAX（↓ 4096 ÷ s ↓，2）－ 1）

s：ステータスファイルのレコード長

v：32 ビットモードの場合：↑（24 ＋ 28 × x ＋ 32 ＋ 112 × y）÷ g ↑
64 ビットモードの場合：↑（32 ＋ 32 × x ＋ 32 ＋ 128 × y）÷ g ↑

w：32 ビットモードの場合：↑（((↓（↑（c ÷ 8）↑＋ 3）÷ 4 ↓）× 4）× z）÷ g ↑
64 ビットモードの場合：↑（((↓（↑（c ÷ 8）↑＋ 7）÷ 8 ↓）× 8）× z）÷ g ↑

x：pd_max_add_dbbuff_shm_no オペランドの値

y：pd_max_add_dbbuff_no オペランドの値

z：pdbuffer オペランドの -i オプション指定総数に，pd_max_add_dbbuff_no オペランドの値を加算した値（pd_dbbuff_modify オペランドに Y 以外を指定した場合は 0 として計算します）

注※ 1　64 ビットモードの場合は 3456 です。

注※ 2　64 ビットモードの場合は 688 です。

### （b） HiRDB/ パラレルサーバの場合

| 種別 | S の計算式 |
|---|---|
| ユニット用 | ↑［2056 ＋ 128 ×｛14 ×（p ＋ q）＋（p ＋ q ＋ r）<br>＋（24 ＋ s × 2 ＋ q × 7 ＋ r × 3 ＋ p × 15 ＋ 2）<br>＋（38 ＋（p ＋ q ＋ r）× 4 ＋ z ＋ d）＋ 3｝］÷ j ↑<br>＋↑ 944 ÷ j ↑＋↑ 4816 ÷ j ↑＋↑ 40 ÷ j ↑<br>＋↑ 308 ÷ j ↑＋↑ 4000 ÷ j ↑＋↑ 43536 ÷ j ↑＋↑ 32 ÷ j ↑<br>**●ユニット内にシステムマネジャがある場合に加算します。**<br>＋↑（32 ＋ 16 ×システム内のユニットの総数）÷ j ↑<br>＋↑（72 ＋ 24 ×（システム内のユニットの総数－ 1））÷ j ↑<br>**●影響分散スタンバイレス型系切り替え機能を使用する場合に加算します。**<br>＋↑｛8192 ＋↑（f × 98 ÷ 8192）↑× 8192 ＋↑（g × 80 ÷ 8192）↑× 8192｝÷ j ↑ |
| フロントエンド<br>サーバ用 | ↑ 128 ÷ j ↑＋↑ 3280 ※ 1 ÷ j ↑＋↑（592 ※ 2 ＋ A × b）÷ j ↑<br>＋↑ 8192 ÷ j ↑＋↑（8192 － 128）÷ j ↑×｛↑（a ＋ C）÷ B ↑－ 1｝<br>＋↑ 8192 ÷ j ↑＋↑ 2048 ÷ j ↑<br>＋↑｛244 ＋ 64 ×（2 × e ＋ 7）｝÷ j ↑＋↑ 96 ÷ j ↑＋↑ 32 ÷ j ↑<br>● pd_log_auto_unload_path **オペランドを指定する場合に加算します。**<br>＋↑ 20848 ÷ j ↑ |
| ディクショナリ<br>サーバ用 | ↑ 128 ÷ j ↑＋↑ 3280 ※ 1 ÷ j ↑＋↑（592 ※ 2 ＋ A × b）÷ j ↑<br>＋↑ 8192 ÷ j ↑＋↑ 8192 ÷ j ↑×｛↑（a ＋ C）÷ B ↑－ 1｝<br>＋↑［↑｛(↑ c ÷ 127 ↑＋ 1）× 2048｝÷ 8192 ↑× 8192］÷ j ↑<br>＋↑ 2048 ÷ j ↑＋↑｛244 ＋ 64 ×（2 × h ＋ 7)｝÷ j ↑＋↑ 96 ÷ j ↑<br>＋↑｛48 ＋ 68 ×（2 × h ＋ 7)｝÷ j ↑<br>＋↑｛12288 ＋↑（f × 98 ÷ 12288）↑× 12288 ＋↑（g × 80 ÷ 12288）↑× 12288｝÷ j ↑<br>＋↑（8 × c ＋ 4）÷ j ↑＋↑ 32 ÷ j ↑<br>＋ H<br>**●ステータスファイルのレコード長＜** 4096 **の場合に加算します。**<br>＋↓ MAX｛3400 ÷ D ＋ 0.7，1｝↓× MAX（↓ 4096 ÷ n ↓，2） |

| 種別 | S の計算式 |
|---|---|
| | ●4096 ≦ステータスファイルのレコード長＜ 12288 の場合に加算します。<br>＋↓ MAX｛(3400 ÷↓（n － 348）÷ 20 ↓）＋ 0.7，1｝↓<br>●12288 ≦ステータスファイルのレコード長の場合に加算します。<br>＋↓ MAX｛(3400 ÷↓（n － 876）÷ 20 ↓）＋ 0.7，1｝↓<br>●系切り替え機能を使用する場合に加算します。<br>＋↑（256 × c ＋ 4096）÷ j ↑<br>● pd_log_auto_unload_path オペランドを指定する場合に加算します。<br>＋↑ 20848 ÷ j ↑<br>● HiRDB External Data Access 機能を使用する場合に加算します。<br>＋↑｛128 ×（v ＋ w ＋ 1)｝÷ j ↑<br>● pd_dbbuff_modify オペランドに Y を指定する場合に加算します。<br>＋ G |
| バックエンド<br>サーバ用 | ↑ 128 ÷ j ↑＋↑ 3280 ※1 ÷ j ↑＋↑（592 ※2 ＋ A × b）÷ j ↑<br>＋↑ 8192 ÷ j ↑＋↑ 8192 ÷ j ↑×｛↑（a ＋ C）÷ B ↑－ 1｝<br>＋↑［↑｛(↑ c ÷ 127 ↑＋ 1）× 2048｝÷ 8192 ↑× 8192］÷ j ↑<br>＋↑ 2048 ÷ j ↑＋↑｛244 ＋ 64 ×（2 × i ＋ 7)｝÷ j ↑＋↑ 96 ÷ j ↑<br>＋↑｛48 ＋ 68 ×（2 × i ＋ 7)｝÷ j ↑＋↑（8 × c ＋ 4）÷ j ↑＋↑ 256 ÷ j ↑<br>＋↑｛12288 ＋↑（f × 98 ÷ 12288）↑× 12288 ＋↑（g × 80 ÷ 12288）↑× 12288｝÷ j ↑※3<br>＋↑ 32 ÷ j ↑＋ H<br><br>●ステータスファイルのレコード長＜ 4096 の場合に加算します。<br>＋↓ MAX｛3400 ÷ D ＋ 0.7，1｝↓× MAX（↓ 4096 ÷ n ↓，2）× k<br>＋↓ 5662310 ÷ D ＋ 0.7 ↓× MAX（↓ 4096 ÷ n ↓，2）× m<br>●4096 ≦ステータスファイルのレコード長＜ 12288 の場合に加算します。<br>＋↓ MAX｛(3400 ÷↓（n － 348）÷ 20 ↓）＋ 0.7，1｝↓× k<br>＋↓（5662310 ÷↓（n － 348）÷ 20 ↓）＋ 0.7 ↓× m<br>●12288 ≦ステータスファイルのレコード長の場合に加算します。<br>＋↓ MAX｛(3400 ÷↓（n － 876）÷ 20 ↓）＋ 0.7，1｝↓× k<br>＋↓（5662310 ÷↓（n － 876）÷ 20 ↓）＋ 0.7 ↓× m<br>●系切り替え機能を使用する場合に加算します。<br>＋↑（256 × y ＋ 4096）÷ j ↑<br><br>● pd_log_auto_unload_path オペランドを指定する場合に加算します。<br>＋↑ 20848 ÷ j ↑<br>● HiRDB External Data Access 機能を使用する場合に加算します。<br>＋↑｛128 ×（t ＋ u ＋ 1)｝÷ j ↑<br>● pd_dbbuff_modify オペランドに Y を指定する場合に加算します。<br>＋ G |

a：pdlogadfg オペランドの指定数

b：pdlogadfg -d spd オペランドの指定数

c：pd_max_rdarea_no オペランドの値

d：3
ただし，1 サーバマシンでの HiRDB/ パラレルサーバの場合は 2

e：pd_max_users オペランドの値＋ 1
ただし，1 サーバマシンでの HiRDB/ パラレルサーバの場合は pd_max_users オペランドの値

f：pdbuffer オペランドの指定数

g：pdbuffer オペランドの -i オプション指定数

h：pd_max_dic_process オペランドの値

i：pd_max_bes_process オペランドの値

j：ステータスファイルのレコード長－ 40

k：サーバ内の RD エリア数≦ 3400 の場合は 1
3401 ≦サーバ内の RD エリア数≦ 6800 の場合は 2
6801 ≦サーバ内の RD エリア数の場合は 3

m：サーバ内の RD エリア数≦ 10200 の場合は 0
10201 ≦サーバ内の RD エリア数≦ 5672510 の場合は 1
5672511 ≦サーバ内の RD エリア数≦ 11334820 の場合は 2
11334821 ≦サーバ内の RD エリア数の場合は 3

n：ステータスファイルのレコード長

p：ユニット内のバックエンドサーバ数

q：ユニット内にディクショナリサーバがある場合は 1，ない場合は 0

r：ユニット内にフロントエンドサーバがある場合は 1，ない場合は 0

s：ユニット内にシステムマネジャがある場合は 1，ない場合は 0

t：このバックエンドサーバが接続している外部サーバ定義の数

u：このバックエンドサーバが接続している外部サーバに対するユーザマッピング定義の数

v：外部サーバ定義の総数

w：ユーザマッピング定義の総数

y：pd_max_rdarea_no オペランドの値

z：次の場合は 1
- ユニット内にシステムマネジャがある場合
- ユニット内にシステムマネジャがない場合で，pd_mlg_msg_log_unit オペランドに local を指定しているとき

ユニット内にシステムマネジャがない場合で，pd_mlg_msg_log_unit オペランドに manager を指定しているか，又は指定を省略しているときは 0

A：736

B：11

C：1

D：↓（n － 348）÷ 20 ↓＋↓ j ÷ 20 ↓×（MAX（↓ 4096 ÷ n ↓，2）－ 1）

G：32 ビットモードの場合：↑（24 ＋ 28 × I ＋ 32 ＋ 112 × J）÷ j ↑
64 ビットモードの場合：↑（32 ＋ 32 × I ＋ 32 ＋ 128 × J）÷ j ↑

H：32 ビットモードの場合：↑（((↓（↑（c ÷ 8）↑＋ 3）÷ 4 ↓）× 4）× K）÷ j ↑
64 ビットモードの場合：↑（((↓（↑（c ÷ 8）↑＋ 7）÷ 8 ↓）× 8）× K）÷ j ↑

I：pd_max_add_dbbuff_shm_no オペランドの値

J：pd_max_add_dbbuff_no オペランドの値

K：pdbuffer オペランドの -i オプション指定総数に，pd_max_add_dbbuff_no オペランドの値を加算した値（pd_dbbuff_modify オペランドに Y 以外を指定した場合は 0 として計算します）

注※ 1　64 ビットモードの場合は 3456 です。

注※ 2　64 ビットモードの場合は 688 です。

注※ 3　影響分散スタンバイレス型系切り替え機能を使用する場合は，加算しないでください。

# 17.4　監査証跡ファイルの容量の見積もり

監査証跡ファイル用の HiRDB ファイルシステムの容量は，次に示す計算式で求めます。

**計算式**

監査証跡ファイル用の HiRDB ファイルシステムの容量（単位：メガバイト）＝ a ＋ 19

a：最も多い場合の監査証跡のデータ量（単位：メガバイト）
↑（Σ {b × c}）÷（1024 × 1024）↑
なお，監査証跡ファイルの入出力エラーが発生した場合や，pdaudswap コマンドで監査証跡ファイルのスワップを実行した場合は，監査証跡ファイルの容量が一杯になる前にスワップします。この場合，監査証跡表へのデータ登録が終了するまで，その空き容量は使用できません。そのため，あらかじめ a の値を 2 倍にしておくことを推奨します。

b：監査証跡のレコードサイズ（単位：バイト）
監査証跡のレコードサイズは，次に示す計算式で求めます。

464 ＋ d ＋ e ＋ f ＋ g ＋ h ＋ i

c：監査証跡のレコード数
監査証跡イベントの種類ごとの記録レコード数です。

d：監査証跡のレコードに出力する SQL 文の長さ（単位：バイト）

pd_aud_sql_source_size の値が 0，又は指定していない場合：0

レコード項目に SQL 文がない場合：0

レコード項目に SQL 文があるが，SQL 文が NULL 値の場合：0

上記以外の場合：
↑ Min（pd_aud_sql_source_size の値，SQL 文の平均長）÷ 4 ↑× 4 ＋ 8

e：監査証跡のレコードに出力する SQL データの長さ（単位：バイト）

pd_aud_sql_data_size の値が 0，又は指定していない場合：0

レコード項目に SQL データがない場合：0

レコード項目に SQL データがあるが，SQL データが NULL 値の場合：0

上記以外の場合：
↑ Min（pd_aud_sql_data_size の値，SQL データの平均長）÷ 4 ↑× 4 ＋ 8

f，g，h：監査証跡のレコードに出力するユーザ付加情報 1，2，3 の長さ（単位：バイト）

レコード項目にユーザ付加情報がない場合，又はユーザ付加情報が NULL 値の場合：0

上記以外の場合：
↑ユーザ付加情報の平均長÷ 4 ↑× 4 ＋ 8

i：監査証跡のレコードに出力する関連製品付加情報の長さ（単位：バイト）

レコード項目に関連製品付加情報がない場合，又は関連製品付加情報が NULL 値の場合※1：0

上記以外の場合：

↑関連製品付加情報の平均長※2 ÷ 4 ↑× 4 ＋ 8

注※ 1

関連製品付加情報が NULL 値になる場合の条件については，関連製品のマニュアルを参照してください。

注※ 2

関連製品付加情報の平均長については，関連製品のマニュアルを参照してください。マニュアルに該当する記載がない場合は，関連製品付加情報の最大長で計算してください。

監査証跡のレコード項目，及び監査証跡ファイルに出力される情報については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

# 18 作業表用ファイルの容量の見積もり

この章では，作業表用ファイルの容量の見積もり方法について説明します。


18.1 **作業表用ファイルの容量の見積もりの概要**

18.2 HiRDB **ファイルシステム領域サイズの見積もり（**pdfmkfs -n **コマンド）**

18.3 **最大ファイル数の見積もり（**pdfmkfs -l **コマンド）**

18.4 **最大増分回数の見積もり（**pdfmkfs -e **コマンド）**

# 18.1 作業表用ファイルの容量の見積もりの概要

ここでは，SQL 文を実行するときに必要とする一時的な情報を格納する**作業表用ファイル**の容量の見積もりについて説明します。

作業表用ファイルとは，次に示す操作をするときに発生する一時的な情報を格納するファイルです。

- SQL 文を実行するとき
- インデクスを一括作成するとき
- インデクスを再作成するとき
- インデクスを再編成するとき
- リバランスユティリティを実行するとき

作業表用ファイルは，HiRDB が HiRDB ファイルシステム領域に作成します。HiRDB 管理者は次に示すことをしてください。

- pdfmkfs **コマンドで，作業表用ファイルを作成する** HiRDB **ファイルシステム領域を初期設定します。**
- HiRDB **システム定義の** pdwork **オペランドで，上記の** HiRDB **ファイルシステム領域の名称を指定します。**

以降，pdfmkfs コマンドのオプションに指定する値の見積もり方法について説明します。pdfmkfs コマンドのオプションと作業表用ファイルの内容の関係を次の表に示します。

**表 18-1 pdfmkfs コマンドのオプションと作業表用ファイルの内容の関係**

| オプション | 内　容 |
|---|---|
| -n | 作業表用ファイルを作成する HiRDB ファイルシステム領域のサイズ |
| -l | HiRDB ファイルシステム領域に作成する HiRDB ファイル（作業表用ファイル）の最大数 |
| -e | HiRDB ファイルシステム領域の最大増分回数 |
| -a | HiRDB ファイルシステム領域を自動的に拡張するかどうか |

# 18.2　HiRDB ファイルシステム領域サイズの見積もり（pdfmkfs -n コマンド）

作業表用ファイルを作成する HiRDB ファイルシステム領域のサイズは，pdfmkfs コマンドの -n オプションで設定します。

HiRDB ファイルシステム領域のサイズは，次に示す計算式で求めます。

**計算式**

HiRDB ファイルシステム領域のサイズ（単位：バイト）＝ A ＋ B

A：

SQL 文が使用する作業表用ファイルの容量です。求め方については，「18.2.1　SQL 文が使用する作業表用ファイルの容量」を参照してください。

B：

データベース作成ユティリティ（pdload），データベース再編成ユティリティ（pdrorg），及びリバランスユティリティ（pdrbal）が使用する作業表用ファイルの容量です。求め方については，「18.2.2　ユティリティが使用する作業表用ファイルの容量」を参照してください。

なお，作業表用ファイルを使用する SQL 文と作業表用ファイルを使用するユティリティの操作を同時に実行しない場合は，A 又は B のどちらか大きい値を HiRDB ファイルシステム領域のサイズとしてください。

**注意事項**

ここで求めた HiRDB ファイルシステム領域のサイズが大き過ぎて，一つの HiRDB ファイルシステム領域に収まらない場合は，複数の HiRDB ファイルシステム領域を pdfmkfs コマンドで初期設定し，HiRDB システム定義の pdwork オペランドに指定してください。このときに注意することを次に示します。

- 各 HiRDB ファイルシステム領域のサイズを同じにしてください。
- 一つの HiRDB ファイルシステム領域のサイズは，一つの作業表（列情報格納用）の容量より大きくしてください。
- HiRDB ファイルシステム領域を分割し過ぎると，未使用領域が複数の HiRDB ファイルシステム領域に分割されます。したがって，領域すべてが有効活用されないため，容量が不足することがあります。
- 1 作業表用ファイルの容量が 2 ギガバイトを超える場合は，pd_large_file_use オペランドに Y を指定，又は指定を省略して HiRDB ファイルシステム領域を作成してください。1 作業表用ファイルは複数の HiRDB ファイルシステム領域に分割できません。

## 18.2.1　SQL 文が使用する作業表用ファイルの容量

SQL 文が使用する作業表用ファイルの容量は次に示す計算式で求めます。

**計算式**

SQL 文が使用する作業表用ファイルの容量（単位：バイト）＝ MAX（ａ，ｂ）× ｃ

a：1SQL 文が使用する作業表用ファイルの容量の最大値
SQL 文ごとに作業表用ファイルの容量を計算します。その中で最も大きい値を a に代入します。「(1)1SQL 文が使用する作業表用ファイルの容量の求め方」を参照して求めてください。

**b**：ASSIGN LIST 文が使用する作業表用ファイルの容量の最大値
ASSIGN LIST 文ごとに作業表用ファイルの容量を計算します。その中で最も大きい値を b に代入します。「(2)ASSIGN LIST 文が使用する作業表用ファイルの容量の求め方」を参照して求めてください。

**c**：pd_max_users オペランドの値
ただし，マルチフロントエンドサーバを使用している場合のバックエンドサーバでは，pd_max_bes_process オペランドの値になります。

## (1) 1SQL 文が使用する作業表用ファイルの容量の求め方

1SQL 文が使用する作業表用ファイルの容量は，次に示す計算式で求めます。

**計算式**

1SQL 文が使用する作業表用ファイルの容量（単位：バイト）＝ a × b ＋ c × d

a：列情報作業表の容量
b：列情報作業表の最大数
c：位置情報作業表の容量
d：位置情報作業表の最大数

### (a) 列情報作業表の容量の求め方

列情報作業表の容量は次に示す計算式で求めます。

**計算式**

列情報作業表の容量（単位：バイト）
＝↑ a ÷ MIN｛↓（b － 48）÷ c ↓，255｝↑× b × 2

a：列情報作業表の行数（表 18-2 から求めてください）

b：作業表ページ長（**計算式** 1 から求めてください）

c：作業表の行長（**計算式** 2 から求めてください）

**計算式** 1

作業表ページ長※＝ MAX｛↑（作業表の行長＋ 48）÷ 2048 ↑× 2048，4096｝

注※　作業表ページ長は，32,768 バイト以下でなければなりません。

**計算式** 2

$$\text{作業表の行長}^{※} = \sum_{i=1}^{n} Ai + 2 \times n + 6$$

Ai：作業表の各列のデータ長（表 18-3 から求めてください）

n：作業表の列数（表 18-3 から求めてください）

注※　作業表の行長は，32,720 バイト以下でなければなりません。

なお，LIMIT 句指定時，（オフセット行数＋ LIMIT 句に指定した行数）の値が 32,768 以上になる場合は，計算式 2 で算出した作業表の行長に 12 を加算してください。ただし，次の場合には 12 を加算する必要はありません。

- 検索対象の表に EX モードで排他が掛かっている場合
- WITHOUT LOCK を指定した場合
- グループ分け高速化機能を指定した場合
- 複数の表を結合する場合

**表 18-2　列情報作業表の行数の求め方**

| SQL 文 | 列情報作業表の行数 |
|---|---|
| SELECT 文 | 検索する個々の表から探索する行数の合計です。ただし，複数の表を結合していて結合結果行数の方が多い場合は，その行数となります。 |
| CREATE INDEX 文 | 表に格納している行数です。ただし，繰返し列に対するインデクスの場合は，インデクス構成列のうち一つの繰返し列の要素の総数となります。 |

**表 18-3　作業表の各列のデータ長及び列数の求め方**

| SQL 文 | n | $A_i$ |
|---|---|---|
| SELECT 文 | **選択式に指定した列数＋** GROUP BY **句に指定した列数＋** ORDER BY **句に指定した列数＋** HAVING **句に指定した列数＋** FOR UPDATE **句を指定した場合は** 1<br>ただし，選択式に ROW を指定している場合は表の全列数 | 左記の列ごとの実際の長さ<br>ただし，長大データ（BLOB），定義長が 256 バイト以上の文字データ（各国・混在文字データも含む），バイナリデータのうち，下記に属さない列又は位置情報列の場合は 12<br>• 結合条件中に指定する列（結合列）<br>• DISTINCT 句指定の選択式<br>• 限定述語の副問合せ選択式中に指定する列<br>• IN 述語の副問合せ選択式中に指定する列<br>• UNION [ALL] 又は EXCEPT [ALL] によって，集合演算対象となっている問合せ指定中の選択式<br>• ORDER BY 句に指定した列 |
| CREATE INDEX 文 | 1（インデクス情報列）＋ 1（位置情報列） | • インデクス情報列はインデクス構成列のデータ長の合計値<br>• 位置情報列は 12 |

注　各列のデータ長については，次に示す表を参照してください。

- 「表 16-1　データ長一覧」
- 「表 16-2　可変長文字列型のデータ長一覧（抽象データ型及び繰返し列を除く）」
- 「表 16-3　可変長文字列型のデータ長一覧（抽象データ型の場合）」
- 「表 16-4　可変長文字列型のデータ長一覧（繰返し列の場合）」

### （b）列情報作業表の最大数の求め方

列情報作業表の最大数の求め方を次の表に示します。

**表 18-4 列情報作業表の最大数の求め方**

| SQL 文 | 列情報作業表の最大数 |
|---|---|
| SELECT 文 | 1. ～ 10. の指定がない場合は，0 になります。<br>1. ～ 10. の指定がある場合は，対応する作業表増加数をすべて加算した値になります。<br>1. **複数の表を結合して検索する場合**<br>作業表増加数（HiRDB/ シングルサーバの場合）＝結合表数＋ 1<br>作業表増加数（HiRDB/ パラレルサーバの場合）＝結合表数× 2<br>ただし，結合キーとなる列にインデクスがあり，制限条件がある場合は 0<br>2. ORDER BY **句を指定する場合**<br>作業表増加数＝ 2<br>ただし，非横分割表で，ORDER BY 句を指定した列にインデクスがある場合は 0<br>3. GROUP BY **句を指定しないで，選択式中に集合関数を含む値式を指定する場合**※<br>作業表増加数＝ 1<br>4. GROUP BY **句を指定する場合**<br>作業表増加数＝ GROUP BY 句指定数× 2<br>5. DISTINCT **句を指定する場合**<br>作業表増加数＝ DISTINCT 句指定数× 2<br>6. UNION［ALL］**句又は** EXCEPT［ALL］**句を指定する場合**<br>作業表増加数（HiRDB/ シングルサーバの場合）＝ UNION［ALL］句又は EXCEPT［ALL］句指定数＋ 2<br>作業表増加数（HiRDB/ パラレルサーバの場合）＝（UNION［ALL］句又は EXCEPT［ALL］句指定数＋ 1）× 2<br>7. FOR UPDATE **句を指定するか又はこのカーソルを使用した更新があり，インデクスを定義した列に検索条件を指定する場合**※<br>作業表増加数＝ 2<br>8. FOR READ ONLY **句を指定する場合**<br>作業表増加数＝ 1<br>9. **副問合せ（限定述語）を指定する場合**<br>増加作業表数＝副問合せ指定数＋（インデクスを定義した列に対する＝ ANY の限定述語の数）＋（インデクスを定義した列に対する IN 述語の副問合せ指定数）＋（インデクスを定義した列に対する＝ SOME の限定述語の数）<br>10. **選択式にウィンドウ関数** COUNT(*) OVER() **を指定する場合**<br>増加作業表数＝選択式にウィンドウ関数を指定した問合せ指定数 |
| CREATE INDEX 文 | 2 |

注※ HiRDB/ パラレルサーバの場合にだけ該当します。

### (c) 位置情報作業表の容量の求め方

位置情報作業表の容量は次に示す計算式で求めます。

**計算式**

位置情報作業表の容量（単位：バイト）＝↑ a ÷ 184※↑× 4096 × 2

注※ 探索条件にインデクス型プラグイン専用関数を指定する場合は 155

a：位置情報作業表の行数

位置情報作業表の行数の求め方を次に示します。

| SQL 文 | 位置情報作業表の行数の求め方 |
|---|---|
| SELECT 文<br>UPDATE 文<br>DELETE 文 | 探索時の条件中にインデクスを定義した列を含む述語が一つの場合，述語が真となる行数になります。二つ以上の場合，次の行数の合計値になります。<br>• 述語を OR 論理演算している場合，少なくとも一つの述語が真となる行数の合計<br>• 述語を AND 論理演算している場合，述語のうち真となる行数が多い方の行数として合算した行数 |

#### (d) 位置情報作業表の最大数の求め方

位置情報作業表の最大数の求め方を次の表に示します。

**表 18-5 位置情報作業表の最大数の求め方**

| SQL 文 | 位置情報作業表の最大数 |
|---|---|
| SELECT 文 | 1. インデクスを定義した複数の列に検索条件を指定する場合<br>2. FOR UPDATE 句を指定するか又はこのカーソルを使用した更新があり，インデクスを定義した列に検索条件を指定する場合※<br>3. 繰返し列インデクスを定義した列に検索条件を指定する場合<br>4. SQL 最適化オプションに「プラグイン提供関数からの一括取得機能」を指定し，探索条件にプラグインインデクスを使用するプラグイン提供関数を指定して検索する場合<br>1 ～ 4 で，探索時に使用するインデクス数＋ 1 になります。 |
| UPDATE 文<br>DELETE 文 | 探索条件中にインデクスを定義した列がある場合，探索時に使用するインデクス数＋ 1 になります。 |

注※ HiRDB/ シングルサーバの場合にだけ該当します。

### (2) ASSIGN LIST 文が使用する作業表用ファイルの容量の求め方

ASSIGN LIST 文が使用する作業表用ファイルの容量は，次に示す計算式で求めます。

**計算式**

ASSIGN LIST 文が使用する作業表用ファイルの容量（単位：バイト）=

$$\sum_{i=1}^{n} (B_i \times 2)$$

n：ASSIGN LIST 文の探索条件中の述語の数

Bi：探索条件中の i 番目の述語を処理する作業表の容量。次に示す計算式で求めます。

$B_i$ ＝↑リストの基になる表で i 番目の述語が真となる行数※÷ 504 ↑× 4096 × 1.5（単位：バイト）

注※ 述語が繰返し列に対する条件の場合，真となる要素の総数となります。

## 18.2.2 ユティリティが使用する作業表用ファイルの容量

インデクスの一括作成，インデクスの再作成，インデクスの再編成，又はリバランスユティリティでデータの再配置をする場合は，次に示す計算式の分の作業表用ファイルが必要になります。

**計算式**

ユティリティが使用する作業表用ファイルの容量（単位：バイト）＝｛(A ＋ B) × 2 × D｝÷ C

A：インデクス作成時に必要な作業表の行数 1

B：インデクス作成時に必要な作業表の行数 2

C：作業表ページ内の行数

D：作業表ページ長

注

- 1 回のユティリティで複数のインデクスの一括作成又は再作成をする場合は，インデクスのキー長が最も長いインデクスについて求めてください。
- インデクスの一括作成又は再作成を同時に実行する場合は，それぞれで必要な作業表用ファイルの容量を求めて加算してください。
- 複数のユティリティを同時実行する場合は，各ユティリティが使用する作業表用ファイルの容量を計算して合計してください。

## (1) インデクス作成時に必要な作業表の行数 1 の求め方

インデクス作成時に必要な作業表の行数 1 は，次に示す計算式で求めます。

**計算式**

作業表の行数 1 ＝↑ c ÷｛↓↑ a ×（100 − b）× 0.01 ↑÷（d ＋ 22）↓｝↑

a：インデクスを格納するユーザ用 RD エリアのページサイズ

b：CREATE INDEX の PCTFREE オペランドで指定する未使用領域の比率

c：データ件数
繰返し列に対するインデクスの場合は，インデクス構成列のうち一つの繰返し列の各行の要素数の合計

d：インデクスのキー長
インデクスのキー長については「表 16-5　インデクスのキー長一覧」を参照してください。なお，データベースに格納されるキー長は 4 バイトバウンダリされるため，**↑キー長÷ 4 ↑× 4** となります。
複数インデクスのキー長は，「表 16-5　インデクスのキー長一覧」を基に全構成列のキー長を加算してください。

## (2) インデクス作成時に必要な作業表の行数 2 の求め方

インデクス作成時に必要な作業表の行数 2 は，次に示す計算式で求めます。

**計算式**

作業表の行数 2 ＝↑ c ÷｛↓↑ a ×（100 − b）× 0.01 ↑÷（d ＋ 14）↓｝↑

a：インデクスを格納するユーザ用 RD エリアのページサイズ

b：CREATE INDEX の PCTFREE オペランドで指定する未使用領域の比率

c：インデクス作成時に必要な作業表の行数 1
（1）で求めた値を代入してください。

d：インデクスのキー長
インデクスのキー長については「表 16-5　インデクスのキー長一覧」を参照してください。なお，データベースに格納されるキー長は 4 バイトバウンダリされるため，**↑キー長÷ 4 ↑× 4** となります。
複数インデクスのキー長は，「表 16-5　インデクスのキー長一覧」を基に全構成列のキー長を加算してください。

### （3）作業表ページ内の行数の求め方

作業表ページ内の行数は，次に示す計算式で求めます。

**計算式**

作業表ページ内の行数＝ MIN｛↓（b － 48）÷ a ↓，255｝

a：作業表の行長（インデクスのキー長＋ 18）
インデクスのキー長については「表 16-5　インデクスのキー長一覧」を参照してください。キー長は **↑キー長÷ 4 ↑× 4** となります。
複数インデクスのキー長は，「表 16-5　インデクスのキー長一覧」を基に全構成列のキー長を加算してください。

b：作業表ページ長
（4）で求めてください。

### （4）作業表ページ長の求め方

作業表ページ長は，次に示す計算式で求めます。

**計算式**

作業表ページ長※＝ MAX｛↑（作業表の行長＋ 48）÷ 2048 ↑× 2048，4096｝

注※　作業表ページ長は，32,768 バイト以下でなければなりません。

a：作業表の行長（インデクスのキー長＋ 18）
インデクスのキー長については「表 16-5　インデクスのキー長一覧」を参照してください。キー長は **↑キー長÷ 4 ↑× 4** となります。
複数インデクスのキー長は，「表 16-5　インデクスのキー長一覧」を基に全構成列のキー長を加算してください。

# 18.3　最大ファイル数の見積もり（pdfmkfs -l コマンド）

HiRDB ファイルシステム領域に作成する作業表用ファイルの最大ファイル数は，pdfmkfs コマンドの -l オプションで設定します。

HiRDB ファイルシステム領域に作成する作業表用ファイルの最大ファイル数は，次に示す計算式で求めます。

**計算式**

最大ファイル数 ＝ MAX（a，b）× c ＋ 20 ＋ 2※

a：1SQL 文が使用する作業表用ファイルの数
SQL 文ごとに作業表用ファイルの数を計算します。その中で最も大きい値を a に代入します。
「(1)1SQL 文が使用する作業表用ファイルの数の求め方」を参照して求めてください。

b：ASSIGN LIST 文が使用する作業表用ファイルの数
ASSIGN LIST 文ごとに作業表用ファイルの数を計算します。その中で最も大きい値を b に代入します。「(2)ASSIGN LIST 文が使用する作業表用ファイルの数の求め方」を参照して求めてください。

c：pd_max_users オペランドの値
ただし，マルチフロントエンドサーバを使用している場合のバックエンドサーバでは，pd_max_bes_process オペランドの値になります。

注※
作業表用ファイルを使用する SQL 文と作業表用ファイルを使用するユティリティ（データベース作成ユティリティ又はデータベース再編成ユティリティ）を同時に実行する場合に加算します。

## (1) 1SQL 文が使用する作業表用ファイルの数の求め方

1SQL 文が使用する作業表用ファイルの数は，次に示す計算式で求めます。

**計算式**

1SQL 文が使用する作業表用ファイルの数＝列情報作業表の最大数＋位置情報作業表の最大数

列情報作業表の最大数，及び位置情報作業表の最大数については，「18.2.1　SQL 文が使用する作業表用ファイルの容量」を参照してください。

## (2) ASSIGN LIST 文が使用する作業表用ファイルの数の求め方

ASSIGN LIST 文が使用する作業表用ファイルの数は，次に示す計算式で求めます。

**計算式**

ASSIGN LIST 文が使用する作業表用ファイルの数＝ ASSIGN LIST 文の探索条件中の述語の数× 2

### (3) 注意

作業表用ファイルを作成する HiRDB ファイルシステム領域を複数設定する場合は，次に示すことに注意してください。

- **ここで求めた値が** 4096 **より大きい場合は，**-l **オプションには** 4096 **を指定してください。**

# 18.4　最大増分回数の見積もり（pdfmkfs -e コマンド）

HiRDB ファイルシステム領域に作成する作業表用ファイルの最大増分回数は，pdfmkfs コマンドの -e オプションで設定します。

HiRDB ファイルシステム領域の最大増分回数は，次に示す計算式で求めます。

**計算式**

最大増分回数＝ MIN（最大ファイル数× 23，60000）

注

最大ファイル数については，「18.3　最大ファイル数の見積もり（pdfmkfs -l コマンド）」の説明を基に算出した値を代入してください。

# 19 ユティリティ実行時の容量の見積もり

この章では，ユティリティ実行時のファイル容量及びメモリ所要量の見積もり方法について説明します。

19.1 **ユティリティ実行時のファイルの容量の見積もり**

19.2 **ユティリティ実行時のメモリ所要量の見積もり**

# 19.1　ユティリティ実行時のファイルの容量の見積もり

## 19.1.1　データベース作成ユティリティ（pdload）実行時のファイルの容量

データベース作成ユティリティ（pdload）で使用するファイルの容量の計算式を次に示します。

| ファイルの種類 | 容量の計算式（単位：バイト） |
|---|---|
| 入力データファイル | h × b |
| インデクス情報ファイル | **B-tree インデクスの場合**<br>（d ＋ y）×（b ＋ e）＋ 512<br>**プラグインインデクスの場合**<br>（12 ＋ q）× p ＋ 1024<br><br>この計算式は，1 インデクス当たりの容量計算式です。インデクスが複数ある場合は，それぞれのインデクスに対して計算してください。 |
| エラー情報ファイル | k × f ＋ s × 200 ＋ m |
| エラー情報ファイル作成用一時ファイル | 次に示す条件の場合は，表格納 RD エリアがあるサーバごとに，**キー重複エラー数×** 8 **＋プラグイン関数が検知したエラー件数×** 200 の容量がワークファイル出力先ディレクトリに必要です。ワークファイル出力先ディレクトリについては，「2.3.3　ワークファイル出力先ディレクトリの作成」を参照してください。<br>• HiRDB/ パラレルサーバでは，入力ファイルがあるサーバと表格納 RD エリアがあるサーバが異なる場合 |
| LOB 入力ファイル | b<br>Σ（LOB データ長＋ 4）i<br>i=1 |
| LOB 中間ファイル | b<br>Σ ｛<br>i=1<br>c<br>Σ（LOB ファイル名称長 ij ＋ 36）＋ 24｝<br>j=1<br>＋ 1024 ＋ c × 84 |
| エラーデータファイル | MIN（f，g）× h |
| 処理結果ファイル | 1500 ＋表格納サーバ数× 500 |
| ワークファイル※ | ［4 ＋ 2 × R ＋ 2 × r ＋ 4 × I × R ＋ {b ÷（-m オプションに指定した経過メッセージ出力間隔の値)}］× 200 |
| ソート用ワークファイル | **条件 1 の場合**<br>インデクス情報ファイルの容量＋ 4 ×（b ＋ e）<br>**条件 2 の場合**<br>{インデクス情報ファイルの容量＋ 4 ×（b ＋ e)} × 2<br><br>• 条件 1<br>sort 文に指定したワークバッファサイズ≧ E のとき<br>• 条件 2<br>sort 文に指定したワークバッファサイズ＜ E のとき<br><br>E：バッファサイズ<br>マニュアル「HiRDB Version 8 コマンドリファレンス」の「データベース作成ユティリティ（pdload)」の「ソート用バッファサイズ」で求めたバッファサイズ |

a：入力行数× LOB 列数

b：入力行数（繰返し列の場合は入力行数×要素数）

c：LOB 列数

d：インデクスのキー長
「表 16-5　インデクスのキー長一覧」を参照してください。ただし，可変長データの場合は単一列でも複数列として扱い，定義長の最大値で計算してください。

e：既存の行数（繰返し列の場合は既存行数×要素数）

f：エラーデータ件数

g：source 文の errdata オペランドで指定する出力行数

h：平均ソースレコード長

k：抽象データ型の列がある場合は 300
ない場合は 120

m：DAT 形式，又は pdrorg で出力したバイナリ形式のファイルの場合は 0
そのほかの場合は（入力ファイルの 1 行のレコード長× 4）

p：インデクス格納 RD エリアを初期化した場合は（b ＋ e）
そのほかの場合は b

q：次に示す値
- LOB 用 RD エリアに格納された抽象データ型の場合は 27
- 定義長 255 バイト以下の抽象データ型の場合は（キー長＋ 2）
- 定義長 256 バイト以上の抽象データ型の場合は 2

代表的な抽象データ型の値を次に示します。
- SGMLTEXT 型の場合は 27
- FREEWORD,GEOMETRY, 及び XML 型の場合は 2

r：LOB 格納 RD エリア数

s：サーバ数

y：キー構成列がすべて固定長の場合は 10
キー構成列に可変長を含む場合は 12

I：インデクス数

R：表，又はインデクスの分割 RD エリア数

注
インデクス情報ファイル及びソート用ワークファイルの容量を算出するとき，インデクス構成列が繰返し列の場合は b 及び e は行数ではなく，行数×要素数となります。

注※
-m オプションでインフォメーションメッセージ出力抑止レベルに lvl2 を指定した場合に出力されます。

## 19.1.2 データベース再編成ユティリティ（pdrorg）実行時のファイルの容量

データベース再編成ユティリティ（pdrorg）で使用するファイルの容量の計算式を次に示します。

| ファイルの種類 | 容量の計算式（単位：バイト） |
|---|---|
| アンロードデータファイル<br>（オプション指定なし） | $\sum_{i=1}^{n}(L_i + P_i) + 1200 + A + B + c \times 96 + D + I + F$ |
| アンロードデータファイル<br>（-W オプション指定時） | **DAT 形式，又は拡張 DAT 形式の場合**<br>$\sum_{i=1}^{c}$（列の最大文字変換長 i [※1] ＋ 1）× n<br>**バイナリ形式で FIX 表の場合**<br>$\sum_{i=1}^{c}$（列データ長 i [※2]）× n<br>**バイナリ形式で非 FIX 表の場合**<br>{ $\sum_{i=1}^{c}$（列データ長 i [※2] ＋ G）＋ 4 ×（c ＋ 1） } × n<br>**固定長文字形式の場合**<br>$\sum_{i=1}^{c}$（列の最大文字変換長 i [※3] ＋ crlf）× n |
| アンロードデータファイル<br>（-j オプション指定時又はスキーマ単位の再編成時）[※4] | $\sum_{i=1}^{n}(L_i + P_i) + \sum_{i=1}^{n}\{\sum_{j=1}^{m}(Oij + 44)\} + 1200 + A + B + c \times 96 + D + I + F$ |
| LOB データのアンロードデータファイル | $\sum_{i=1}^{n}\{\sum_{j=1}^{m}(Oij + 44)\} + 1200 + A + B + c \times 96 + D + I + F$ |
| インデクス情報ファイル | **B-tree インデクスの場合**<br>（K ＋ p）× n ＋ 512<br>**プラグインインデクスの場合**<br>（12 ＋ X）× n ＋ 1024<br><br>この計算式は，1 インデクス当たりの容量計算式です。インデクスが複数ある場合は，それぞれのインデクスに対して計算してください。 |
| 処理結果ファイル | 1700 ＋表格納サーバ数× 500 ＋スキーマ内表数× 1000 ＋スキーマ内総格納 RD エリア数× 100 |
| ワークファイル[※5] | ［8 ＋ 2 × S ＋ 2 ×｛n ÷（-m オプションに指定した経過メッセージ出力間隔の値）｝＋ 3 × R ＋ 4 × J × R］× 200 |

| ファイルの種類 | 容量の計算式（単位：バイト） |
|---|---|
| ソート用ワークファイル | **条件 1 の場合**<br>インデクス情報ファイルの容量＋ 4 × n<br>**条件 2 の場合**<br>{インデクス情報ファイルの容量＋ 4 × n} × 2<br><br>• 条件 1<br>sort 文に指定したワークバッファサイズ≧ E のとき<br>• 条件 2<br>sort 文に指定したワークバッファサイズ＜ E のとき<br><br>E：バッファサイズ<br>マニュアル「HiRDB Version 8 コマンドリファレンス」の「データベース再編成ユティリティ（pdrorg）」の「ソート用バッファサイズ」で求めたバッファサイズ |

A：キーレンジ分割の場合：48 ＋分割条件数× 284
ハッシュ分割の場合：40 ＋ a × 60
マトリクス分割（境界値指定のキーレンジ分割とハッシュ分割の組み合わせ）の場合：
48 ＋（分割条件数× 284）＋（40 ＋ a × 60）

B：n × 36（FIX 表の場合）又は（44 ＋ c × 4）× n（非 FIX 表の場合）

D：16 ＋（LOB 列数× a × 80）
D は LOB 列がある場合に加算します。

F：次の値を代入してください。

$$\sum_{i=1}^{d} \{(\text{列 i のプラグインが提供する抽象データ型の属性数} \times 84) + (\text{列 i のプラグインが提供する抽象データ型の LOB 属性数} \times a \times 72)\} + 64 + \sum_{i=1}^{d} (84 + \text{逆生成関数数 i} \times 60)$$

G：列 I に対する逆生成関数の返却値が BLOB の属性数× 4

I：136 ＋インデクス分割数× 60
インデクス引き上げ時に加算します。

J：インデクス数

K：インデクスのキー長
「表 16-5　インデクスのキー長一覧」を参照してください。ただし，可変長データの場合は単一列でも複数列として扱い，定義長の最大値で計算してください。

$L_i$：行の実長
各列のデータ長を合計して，行長を求めます。列のデータ型が BLOB の場合，列長は 16 で計算してください。列のデータ型が，プラグインが提供した抽象データ型の場合は列長は 2 で計算してください。

$O_{ij}$：LOB データ長

$P_i$：プラグインが提供する抽象データ型のデータ長

R：表又はインデクスの分割 RD エリア数

S：表格納サーバ数

X：次に示す値
- LOB 用 RD エリアに格納された抽象データ型の場合は 27
- 定義長 255 バイト以下の抽象データ型の場合は（キー長＋ 2）
- 定義長 256 バイト以上の抽象データ型の場合は 2

代表的な抽象データ型の値を次に示します。
- SGMLTEXT 型の場合は 27
- FREEWORD,GEOMETRY, 及び XML 型の場合は 2

a：分割 RD エリア数

c：列定義数

d：プラグインが提供する抽象データ型が定義された列数

m：LOB 列数

n：行数（繰返し列の場合は行数×要素数）

p：キー構成列がすべて固定長の場合は 10，キー構成列に可変長を含む場合は 12

crlf：-W オプションに cr 又は crlf を指定した場合に加算する改行文字の長さ
次の表から改行文字の長さを求めます。

| -W オプション指定値 | | 加算値 |
|---|---|---|
| -W dat 又は -W extdat | ,cr | 1 |
| | ,crlf | 2 |
| | 上記の指定なし | 1 |
| -W fixtext | ,cr | 1 |
| | ,crlf | 2 |
| | 上記の指定なし | 0 |

注
インデクス情報ファイル及びソート用ワークファイルの容量を算出するとき，インデクス構成列が繰返し列の場合は「リロードする行数」及び n は行数ではなく，行数×要素数となります。

注※ 1
DAT 形式（-W dat）又は拡張 DAT 形式（-W extdat）の場合の列の最大文字変換長を次の表に示します。

**表 19-1 列の最大文字変換長（DAT 形式又は拡張 DAT 形式の場合）**

| データ型 | | 最大文字変換長（バイト） |
|---|---|---|
| 数データ | INTEGER | 11 |
| | SMALLINT | 11 |
| | DECIMAL | 40 |
| | FLOAT | 23 |
| | SMALLFLT | 23 |

| データ型 | | 最大文字変換長（バイト） |
|---|---|---|
| 文字データ※1 | CHARACTER | 定義長＋2※2 |
| | VARCHAR | 実長＋2※2 |
| 混在文字データ※1 | MCHAR | 定義長＋2※2 |
| | MVARCHAR | 実長＋2※2 |
| 各国文字データ※1 | NCHAR | 定義長×2＋2※2 |
| | NVARCHAR | 実長＋2※2 |
| 日付データ | DATE | 10 |
| 時刻 | TIME | 8 |
| 日間隔データ | INTERVAL YEAR TO DAY | 9 |
| 時間隔データ | INTERVAL HOUR TO SECOND | 7 |
| 時刻印データ | TIMESTAMP | 19<br>小数秒けた数が0でなければ，小数秒けた数＋1を加算してください。 |
| バイナリデータ※1 | BINARY | 実長＋2※2 |

注※1

拡張 DAT 形式の場合，データ中に「”」があると，「”」の数分，文字変換長が長くなります。

注※2

データの長さに囲み文字数を2バイト加算しています。

ただし，-W dat 又は -W extdat 指定時，オペランドに sup を指定した場合の列の最大文字変換長は次に示すとおりです。なお，表中の実長とは，後方の連続スペースを圧縮した残りの長さです。スペースを圧縮した場合の出力形式については，マニュアル「HiRDB Version 8 コマンドリファレンス」のデータベース再編成ユティリティ（pdrorg）の -W オプションの説明を参照してください。

| データ型 | | 最大文字変換長（バイト） |
|---|---|---|
| 文字データ | CHARACTER | 実長＋2 |
| 混在文字データ | MCHAR | 実長＋2 |
| 各国文字データ | NCHAR | 実長＋2 |

注※2

データ長については次を参照してください。

- 「表 16-1　データ長一覧」
- 「表 16-2　可変長文字列型のデータ長一覧（抽象データ型及び繰返し列を除く）」
- 「表 16-3　可変長文字列型のデータ長一覧（抽象データ型の場合）」
- 「表 16-4　可変長文字列型のデータ長一覧（繰返し列の場合）」

注※3

固定長文字形式（-W fixtext）の場合の列の最大文字変換長を次の表に示します。

表 19-2　列の最大文字変換長（固定長文字形式の場合）

| データ型 | | 最大文字変換長（バイト） | |
|---|---|---|---|
| 数データ | INTEGER | 11 | |
| | SMALLINT | 6 | |
| | DECIMAL | けた数＋ 2 | |
| | FLOAT | 23 | |
| | SMALLFLT | 23 | |
| 文字データ | CHARACTER<br>VARCHAR | 定義長 | fixtext_option に enclose オペランドを指定した場合は出力長に 2 を加算してください。 |
| 混在文字データ | MCHAR<br>MVARCHAR | 定義長 | |
| 各国文字データ | NCHAR<br>NVARCHAR | 定義長× 2 | |
| 日付データ | DATE | 10 | |
| 時刻 | TIME | 8 | |
| 日間隔データ | INTERVAL YEAR TO DAY | 10 | |
| 時間隔データ | INTERVAL HOUR TO SECOUND | 8 | |
| 時刻印データ | TIMESTAMP | 小数部<br>0:19　2:22　4:24　6:26 | |
| 長大データ | BLOB | 0 | |
| バイナリデータ | BINARY | 0 | |
| 抽象データ型 | ADT | 0 | |

注※ 4

スキーマ単位の再編成（アンロードも含む）をする場合は，表ごとに求めた値を合計してください。

注※ 5

-m オプションでインフォメーションメッセージ出力抑止レベルに lvl2 を指定した場合に出力されます。

## 19.1.3　統計解析ユティリティ（pdstedit）実行時のファイルの容量

統計解析ユティリティ（pdstedit）で使用するファイルの容量の計算式を次に示します。

| ファイルの種類 | | 容量の計算式（単位：バイト） |
|---|---|---|
| ワーク用ファイル | システムの稼働に関する統計情報 | 4096 ×取得回数※× 2 |
| | サーバごとのシステムの稼働に関する統計情報 | 4096 ×取得回数※×サーバ数× 2 |
| | UAP に関する統計情報 | 800 ×実行 UAP 数又は実行トランザクション数× 2 |
| | SQL に関する統計情報 | 512 ×実行 SQL 数× 2 |
| | SQL 静的最適化に関する統計情報 | 512 × SQL オブジェクトキャッシュでミスヒットした回数 |
| | SQL 動的最適化に関する統計情報 | 8192 × OPEN 又は EXECUTE で発行した SELECT 文の数（INSERT SELECT も含む） |

| ファイルの種類 | | 容量の計算式（単位：バイト） |
|---|---|---|
| | SQL オブジェクト実行に関する統計情報 | 512 ×実行 SQL 数×サーバ数 |
| | SQL オブジェクト転送に関する統計情報 | 256 ×実行 SQL 数×サーバ数 |
| | SQL 文の履歴に関する統計情報 | （1024 ＋平均的な SQL 長）×実行 SQL 数 |
| | CONNECT/DISCONNECT に関する統計情報 | 256 × CONNECT 及び DISCONNECT 回数 |
| | グローバルバッファに関する統計情報 | 512 ×シンクポイント発生回数× 2 |
| | データベース操作に関する HiRDB ファイルの統計情報 | 512 ×シンクポイント発生回数× 2 |
| | デファードライト処理に関する統計情報 | 512 ×デファードライト発生回数× 2 |
| | インデクスに関する統計情報（入力：STJ） | 128 ×シンクポイント発生回数× 2 |
| | インデクスに関する統計情報（入力：FJ） | 128 ×ページスプリット発生回数× 2 |
| | 外部サーバの稼働に関する統計情報 | 512 ×外部サーバにアクセスした実行トランザクション数 |
| | 外部サーバの利用状況に関する統計情報 | 512 ×外部サーバに対する実行 SQL 数 |
| ソート用ワークファイル | 解析対象とした上記のワーク用ファイルのソート用作業領域 | 解析対象とした上記のワーク用ファイルのファイル容量の最大値 |
| DAT 形式ファイル | システムの稼働に関する統計情報 | 3404 ×取得回数※ |
| | サーバごとのシステムの稼働に関する統計情報 | 3262 ×取得回数※×サーバ数 |
| | UAP に関する統計情報 | 1410 ×実行 UAP 数又は実行トランザクション数 |
| | SQL に関する統計情報 | 447 ×実行 SQL 数 |
| | SQL 静的最適化に関する統計情報 | 646 × SQL オブジェクトキャッシュでミスヒットした回数 |
| | SQL 動的最適化に関する統計情報 | 380 × OPEN 又は EXECUTE で発行した SELECT 文の数（INSERT SELECT も含む） |
| | SQL オブジェクト実行に関する統計情報 | 520 ×実行 SQL 数×サーバ数 |
| | SQL オブジェクト転送に関する統計情報 | 389 ×実行 SQL 数×サーバ数 |
| | SQL 文の履歴に関する統計情報 | （240 ＋平均的な SQL 長）×実行 SQL 数 |
| | CONNECT/DISCONNECT に関する統計情報 | 278 × CONNECT 及び DISCONNECT 回数 |
| | グローバルバッファに関する統計情報 | 567 ×シンクポイント発生回数 |
| | データベース操作に関する HiRDB ファイルの統計情報 | 356 ×シンクポイント発生回数 |
| | デファードライト処理に関する統計情報 | 300 ×デファードライト発生回数 |
| | 外部サーバの稼働に関する統計情報 | 374 ×外部サーバにアクセスした実行トランザクション数 |
| | 外部サーバの利用状況に関する統計情報 | 389 ×外部サーバに対する実行 SQL 数 |

注※

取得回数＝↓（pdstend コマンドの入力時刻－ pdstbegin コマンドの入力時刻）÷ -m オプションで指定した時間間隔↓

### 19.1.4 データベース状態解析ユティリティ（pddbst）実行時のファイルの容量

データベース状態解析ユティリティ（pddbst）で使用するファイルの容量の計算式を次に示します。

| ファイルの種類 | | 容量の計算式（単位：キロバイト） |
|---|---|---|
| ワーク用ファイル | RD エリア単位の物理的解析 | 1 ＋ 4.1 × a |
| | RD エリア単位の論理的解析 | $1 + 4.1 \times \sum_{i=1}^{a}$（RD エリア内の表数 i ＋インデクス数 i ＋ LOB 用 RD エリア数 i） |
| | 状態解析結果蓄積又は再編成時期予測※ | $1 + 4.1 \times \sum_{i=1}^{a}$（RD エリア内の表数 i ＋インデクス数 i ＋ LOB 用 RD エリア数 i） |
| | 表単位の状態解析 | 1 ＋ 4.1 ×（格納 RD エリア数） |
| | インデクス単位の状態解析 | 1 ＋ 4.1 ×（格納 RD エリア数） |
| | クラスタキーの状態解析 | 1 ＋ 4.1 ×（格納 RD エリア数） |
| ソート用ワークファイル | 上記のワーク用ファイルのソート用作業領域 | 上記の計算式で求めた値× 2 |

注※

pddbst -r ALL 指定時は，ユーザ用 RD エリア内だけでなく，ディクショナリ用 RD エリア内の資源数を加算する必要があります。また，分割した表やインデクスについては RD エリアごとに数を加算します。

a：解析対象の RD エリア数

### 19.1.5 データベース複写ユティリティ（pdcopy）実行時のファイルの容量

データベース複写ユティリティ（pdcopy）で使用するファイルの容量の計算式を次に示します。

| ファイルの種類 | 容量の計算式（単位：バイト） |
|---|---|
| バックアップファイル※<br>フルバックアップファイル※ | $\sum_{i=1}^{a} \{28 \times c_i + (d_i + 28) \times (e_i + q_i)\}$ ＋ 88 × a ＋ 220 × b |
| 差分バックアップファイル※ | w×フルバックアップファイルの容量 |
| 差分バックアップ管理ファイル | (1 ＋ j ＋ m) × 32768 |
| ログポイント情報ファイル | 1024 |

注※

バックアップファイルの容量が 2 ギガバイトを超える場合は，次に示す対処をしてください。

- 2 ギガバイト以下のパーティションを複数作成して，バックアップファイルを複数指定してください。

a：バックアップ対象 RD エリア数

b：バックアップ対象 RD エリアの HiRDB ファイルの総数

$c_i$：バックアップ対象 RD エリアの未使用ページ数

システム構築前に見積もる場合は 0 と仮定してください。

- ユーザ用 RD エリアの場合
  データベース状態解析ユティリティ（pddbst コマンド）の RD エリア単位の状態解析（物理的解析）を実行して求めてください。ユティリティの実行結果の「RD エリア内のページの情報」の「総ページ数－使用ページ数」の値です。
- ユーザ LOB 用 RD エリアの場合
  データベース状態解析ユティリティ（pddbst コマンド）の RD エリア単位の状態解析（物理的解析）を実行して求めてください。ユティリティの実行結果の「RD エリア内のセグメントの情報」の「総セグメント数－使用セグメント数」の値です。

$d_i$：バックアップ対象 RD エリアのページ長

$e_i$：バックアップ対象 RD エリアの使用ページ数

システム構築前に見積もる場合は，（バックアップ対象 RD エリアのセグメント数×セグメントサイズ）を仮定して計算してください。

- ユーザ用 RD エリアの場合
  データベース状態解析ユティリティ（pddbst コマンド）の RD エリア単位の状態解析（物理的解析）を実行して求めてください。ユティリティの実行結果の「RD エリア内のページの情報」の「使用ページ数」の値です。
- ユーザ LOB 用 RD エリアの場合
  データベース状態解析ユティリティ（pddbst コマンド）の RD エリア単位の状態解析（物理的解析）を実行して求めてください。ユティリティの実行結果の「RD エリア内のセグメントの情報」の「使用セグメント数」の値です。

g：-b オプションに指定するバックアップファイルの名称長（バイト）

複数のバックアップファイルを指定する場合は，合計の名称長になります。

h：-b オプションに指定するバックアップファイルの数

j：↑（512 ＋ 128 × a）÷ 32700 ↑

k：差分バックアップの継続回数

m：↑ {↑（256 ＋ 128 × a ＋ g ＋ 8 × h）÷ 256 ↑× k} ÷ 100 ↑

$q_i$：バックアップ対象 RD エリアのディレクトリページ数

- ユーザ用 RD エリアの場合
  6 ×（ti ＋ 1）＋ 2 ×↑（20480 ÷ di）↑＋ {↑（si ÷ ui）↑＋↑（si ÷ vi）↑＋ 2 × ti}
- ユーザ LOB 用 RD エリアの場合
  7 ＋ 3 ×（ti － 1）＋ {↑（si ÷ 64000）↑＋ ti} × 96

ri：バックアップ対象 RD エリアのセグメントサイズ

si：バックアップ対象 RD エリアの全セグメント数

データベース初期設定ユティリティ（pdinit コマンド）又はデータベース構成変更ユティリティ（pdmod コマンド）の create rdarea 文で指定する HiRDB ファイルのセグメント数の合計です。RD エリアの自動増分を指定している場合は，増分したセグメント数を加算してください。

ti：バックアップ対象 RD エリアの HiRDB ファイル数

ui：↓｛di − 20｝÷｛(↑ ri ÷ 32 ↑× 8）＋ 56｝↓

vi：↑（125 × di）÷（16 × ui）↑× ui

w：バックアップ対象 RD エリアの全ページに対する更新ページの比率
w の算出式（ユーザ用 RD エリア）は次に示す計算式から求めます。

$$w = \left\{ \sum_{i=1}^{a} Xi \div \sum_{i=1}^{a} e_i \right\} \times 1.2$$

Xi：バックアップ対象 RD エリアの更新ページ数
更新ページ数とは，差分の基準となる前回の差分バックアップ取得時から現在までの間に更新されたページの数のことです。更新ページ数は更新 SQL の種類又は更新件数などから，バックアップ対象 RD エリアに格納されている表及びインデクスについて，次に示す条件に従って計算してください。

- **INSERT の場合**
「16.1　ユーザ用 RD エリアの容量の見積もり」を参照して，インサート件数から格納ページ数を計算して Xi に加算してください。このとき，PCTFREE は 0 で計算してください。
- **DELETE の場合**
次に示す条件に応じて計算した値を Xi に加算してください。

| 条件 | | | Xi に加算する値 |
|---|---|---|---|
| 表 | 行長＜ $d_i$ | 削除行が表全体に分散する場合 | MIN（削除行数，表の使用中ページ数） |
| | | 削除行が一部のページに集中する場合 | MIN（削除行数÷ 1 ページで格納できる行数，表の使用中ページ数） |
| | 行長＞ $d_i$ | | MIN（削除行数× 1 行格納するのに必要なページ数，表の使用中ページ数） |
| インデクス | 更新されるキーがインデクス全体に分散する場合 | | MIN（更新キー数＋ α，インデクスの使用中ページ数） |
| | 更新されるキーが一部のページに集中する場合 | | MIN（更新キー数÷ 1 ページで格納できるキー数＋ α，インデクスの使用中ページ数） |

α ：201 以上重複するキーの数

- **UPDATE の場合**
次に示す条件に応じて計算した値を Xi に加算してください。

| 条件 | | | Xi に加算する値 |
|---|---|---|---|
| 表 | 更新列長＜ $d_i$ | 更新行が表全体に分散する場合 | MIN（更新行数，表の使用中ページ数） |
| | | 更新行が一部のページに集中する場合 | MIN（更新行数÷１ページで格納できる行数，表の使用中ページ数） |
| | 更新列長＞ $d_i$ | | MIN（更新行数×更新列を格納するのに必要なページ数，表の使用中ページ数） |
| インデクス | 更新されるキーがインデクス全体に分散する場合 | | MIN（更新キー数×２＋ α ×２，インデクスの使用中ページ数） |
| | 更新されるキーが一部のページに集中する場合 | | MIN（更新キー数×２÷１ページで格納できるキー数＋ α ×２，インデクスの使用中ページ数） |

α ：201 以上重複するキーの数

- **再編成を行った場合**
  再編成を行った表又はインデクスのうち，バックアップ対象 RD エリアに格納されているものについて β を計算してください。
  **β ＝再編成を行った表又はインデクスの使用中ページ数＋再編成を行った表又はインデクスの使用中セグメント数÷** ui **＋再編成を行った表又はインデクスの使用中セグメント数÷** vi
  β を再編成した表又はインデクスの数だけ計算して Xi に加算してください。
- PURGE **の場合**
  PURGE を行った表又はインデクスのうち，バックアップ対象 RD エリアに格納されているものについて γ を計算してください。
  **γ ＝** PURGE **を行った表又はインデクスの使用中セグメント数÷** ui **＋** PURGE **を行った表又はインデクスの使用中セグメント数÷** vi
  γ を再編成した表又はインデクスの数だけ計算して Xi に加算してください。

## 19.1.6　ディクショナリ搬出入ユティリティ（pdexp）実行時のファイルの容量

ディクショナリ搬出入ユティリティ（pdexp）で使用するファイルの容量の計算式を次に示します。

| ファイルの種類 | | 容量の計算式（単位：キロバイト） |
|---|---|---|
| 搬出ファイル※ | 実表の場合 | $0.8 + 0.5 \times CMN + \sum_{n=1}^{DEF}(0.1 + DSn) + 0.1 \times LRD + 0.6 \times DIV + 0.7 \times REF + \sum_{n=1}^{CHK}(0.1 + CSn) + \sum_{n=1}^{IDX}(0.8 + 0.1 \times IRn)$ |
| | ビュー表の場合 | $0.5 + 0.4 \times CMN + e$ |
| | プロシジャの場合 | $1.0 + 0.1 \times g + h$ |

CHK：検査制約数（$0 \leqq CHK \leqq 254$）

CMN：表の列数（1 ≦ CMN ≦ 30,000）

CSn：n 番目の検査制約の検索条件サイズ（0 ≦ CSn ≦ 2,000,000）

DEF：デフォルト値定義列数（0 ≦ DEF ≦ 30,000）

DIV：分割条件数（0 ≦ DIV ≦ 3,000）

DSn：n 番目のデフォルト列のデフォルト値サイズ（1 ≦ DSn ≦ 64,003）

IDX：インデクス数（0 ≦ IDX ≦ 254）

IRn：n 番目のインデクスの格納用 RD エリア数（0 ≦ IRn ≦ 3,000）

LRD：LOB 用 RD エリア数（0 ≦ LRD ≦ 1,024）

REF：参照制約数（0 ≦ REF ≦ 255）

e：ビュー表定義時のソース長（単位：キロバイト）

g：搬出するストアドプロシジャが使用するリソース数
SQL_ROUTINES 表の N_RESOURCE 列の値です。

h：ストアドプロシジャのソース長（単位：キロバイト）
SQL_ROUTINES 表の SOURCE_SIZE 列の値です。

注※
複数の表を搬出する場合，各表に対して上記の計算をしてください。その合計値が搬出ファイルの容量となります。

## 19.1.7 最適化情報収集ユティリティ（pdgetcst）実行時のファイルの容量

最適化情報収集ユティリティ（pdgetcst）で使用するファイルの容量の計算式を次に示します。

| ファイルの種類 | 容量の計算式（単位：バイト） |
|---|---|
| 最適化情報パラメタファイル | 162 ＋ 405 × a ＋ 567 × b |
| 出力結果ファイル | **検索による最適化情報収集時（-c lvl1 指定時）**<br>202 ＋ 131 × e<br>**検索による最適化情報収集時（-c lvl2 指定時）**<br>370 ＋ 561 × c ＋ 196 × d<br>**最適化情報パラメタファイルによる最適化情報登録時**<br>370 ＋ 235 × c ＋ 387 × f |

a：指定インデクス数

b：指定列数

c：表に定義してあるインデクスの数

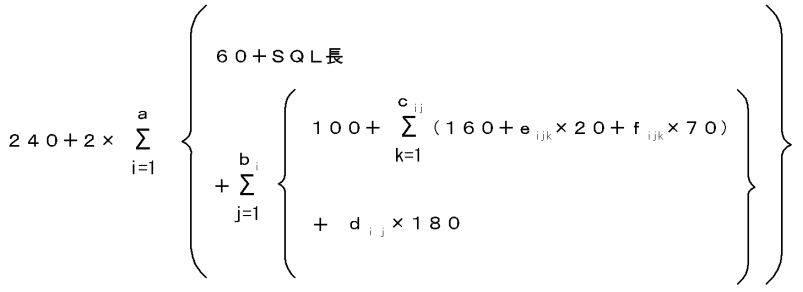
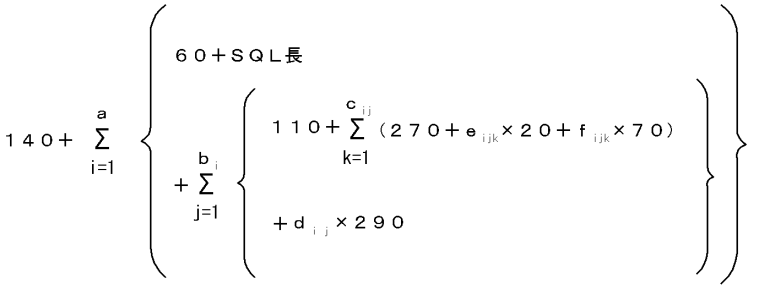
d：区間数

e：表数

f：表の列数

## 19.1.8 アクセスパス表示ユティリティ（pdvwopt）実行時のファイルの容量

アクセスパス表示ユティリティ（pdvwopt）で使用するファイルの容量の計算式を次に示します。

| ファイルの種類 | 容量の計算式（単位：バイト） |
|---|---|
| アクセスパス情報ファイル | ・HiRDB/シングルサーバの場合<br>  <br>・HiRDB/パラレルサーバの場合<br>  |



a：検索系 SQL 数

$b_i$：SQL 中の問合せ数

$c_{ij}$：問合せ中の表数

$d_{ij}$：問合せ中の結合処理数

$e_{ijk}$：表の格納 RD エリア数

$f_{ijk}$：表のインデクス定義数

## 19.1.9 リバランスユティリティ（pdrbal）実行時のファイルの容量

リバランスユティリティ（pdrbal）で使用するファイルの容量の計算式を次に示します。

| ファイルの種類 | 容量の計算式（単位：バイト） |
|---|---|
| インデクス情報ファイル | B-tree **インデクスの場合**<br>（K ＋ d）× N ＋ 512<br>**プラグインインデクスの場合**<br>（12 × Y）× N ＋ 1024<br><br>この計算式は，1 インデクス当たりの容量計算式です。インデクスが複数ある場合は，それぞれのインデクスに対して計算してください。 |
| ワークファイル※1 | （3 ＋ 4 × I × R）× 200 |
| ソート用ワークファイル※2 | **条件 1 の場合**<br>インデクス情報ファイルの容量＋ 4 × N<br>**条件 2 の場合**<br>{インデクス情報ファイルの容量＋ 4 × N} × 2<br><br>• 条件 1<br>sort 文に指定したワークバッファサイズ≧ E のとき<br>• 条件 2<br>sort 文に指定したワークバッファサイズ＜ E のとき<br><br>E：バッファサイズ<br>マニュアル「HiRDB Version 8 コマンドリファレンス」の「リバランスユティリティ（pdrbal）」の「ソート用バッファサイズ」で求めたバッファサイズ |
| 実行結果出力ファイル | 1000 ＋表格納 RD エリア数× 200 |

d：キー構成列がすべて固定長の場合は 10，キー構成列に可変長を含む場合は 12

I：インデクス数

K：インデクスのキー長
「表 16-5　インデクスのキー長一覧」を参照してください。ただし，可変長データの場合は単一列でも複数列として扱い，定義長の最大値で計算してください。

N：リバランス処理によって移動する行数（繰返し列の場合は行数×要素数）

R：表又はインデクスの分割 RD エリア数

Y：次に示す値
- LOB 用 RD エリアに格納された抽象データ型の場合は 27
- 定義長 255 バイト以下の抽象データ型の場合は（キー長＋ 2）
- 定義長 256 バイト以上の抽象データ型の場合は 2

代表的な抽象データ型の値を次に示します。
- SGMLTEXT 型の場合は 27
- FREEWORD,GEOMETRY, 及び XML 型の場合は 2

注※ 1
-m オプションに lvl2 を指定した場合に出力されます。

注※ 2
プラグインインデクスの場合，このファイルは不要です。

## 19.1.10 整合性チェックユティリティ（pdconstck）実行時のファイルの容量

整合性チェックユティリティ（pdconstck）で使用するファイルの容量の計算式を次に示します。

| ファイルの種類 | 容量の計算式（単位：バイト） |
|---|---|
| 処理結果ファイル | -k set/release **の場合**<br>560 ＋（REF ＋ 1）× 70 ＋（CHK ＋ 1）× 70<br>-k check **の場合**<br>700 ＋（REF ＋ 1）× 70 ＋（CHK ＋ 1）× 70<br>REF<br>＋ Σ（490 ＋（RCn × 70）<br>n=1<br>GEN ROW<br>＋ Σ（Σ（RCnml × 70）））<br>m=1 l=1<br>CHK<br>＋ Σ（490 ＋（CCn × 70）<br>n=1<br>GEN ROW<br>＋ Σ（Σ（CCnml × 70）））<br>m=1 l=1 |

REF：表に定義されている参照制約の制約数

CHK：表に定義されている検査制約の制約数

RC：参照制約の外部キー構成列数

CC：検査制約の探索条件中の列数

GEN：1

ROW：制約違反となったキー値の出力数の上限値（-w オプションで指定した値）

# 19.2　ユティリティ実行時のメモリ所要量の見積もり

## 19.2.1　データベース初期設定ユティリティ（pdinit）実行時のメモリ所要量

データベース初期設定ユティリティ（pdinit）実行時のメモリ所要量は，次に示す計算式で求めます。

### （1）HiRDB/シングルサーバの場合

| 条件 | メモリ所要量の計算式（単位：キロバイト） |
|---|---|
| 32 ビットモードの場合 | ↑{<br>{61440 ×（140 × a ＋ 20 × b ＋ c)} ÷ 61432 ＋（6004 × d）÷ 500<br>＋（36008 × a）÷ 1000 ＋ 468 × e ＋ 404399<br>}↑÷ 1024 ＋ 267 |
| 64 ビットモードの場合 | ↑{<br>{61448 ×（144 × a ＋ 24 × b ＋ c)} ÷ 61436 ＋（8004 × d）÷ 500<br>＋（36016 × a）÷ 1000 ＋ 468 × e ＋ 406399<br>}↑÷ 1024 ＋ 267 |

a：RD エリアの総数

b：全 RD エリアの HiRDB ファイル数

c：全 HiRDB ファイルの名称長の合計

d：認可識別子の総数

e：データディクショナリ用 RD エリア数

### （2）HiRDB/パラレルサーバの場合

| 条件 | | メモリ所要量の計算式（単位：キロバイト） |
|---|---|---|
| 32 ビットモードの場合 | DS | ↑{<br>{61448 ×（144 × a ＋ 24 × b ＋ c)} ÷ 61436 ＋（8004 × d）÷ 500<br>＋（36016 × a）÷ 1000 ＋ 468 × e ＋ 406399 ＋ 348 × f ＋ 344 × g<br>}↑÷ 1024 ＋ 268 |
| | BES | ↑（4 × b ＋ 246255）÷ 1024 ↑＋ 268 |
| | MGR | 10 |
| 64 ビットモードの場合 | DS | ↑{<br>{61448 ×（144 × a ＋ 24 × b ＋ c)}÷ 61436 ＋（8004 × d）÷ 500<br>＋（36016 × a）÷ 1000 ＋ 468 × e ＋ 406399 ＋ 348 × f ＋ 344 × g<br>}↑÷ 1024 ＋ 268 |
| | BES | ↑（4 × b ＋ 245744）÷ 1024 ↑＋ 268 |
| | MGR | 10 |

a：RD エリアの総数

b：全 RD エリアの HiRDB ファイル数

c：全 HiRDB ファイルの名称長の合計

d：認可識別子の総数

e：データディクショナリ用 RD エリア数

f：バックエンドサーバの総数

g：(144 × a ＋ 24 × b ＋ c) ÷ 7780 を各バックエンドサーバごとに計算した総計

## 19.2.2　データベース定義ユティリティ（pddef）実行時のメモリ所要量

データベース定義ユティリティ（pddef）実行時のメモリ所要量は，次に示す計算式で求めます。

| 条件 | メモリ所要量の計算式（単位：キロバイト） |
|---|---|
| HiRDB/ シングルサーバの場合 | 32 ビットモードの場合：1956<br>64 ビットモードの場合：1957 |
| HiRDB/ パラレルサーバの場合 | |

## 19.2.3　データベース作成ユティリティ（pdload）実行時のメモリ所要量

データベース作成ユティリティ（pdload）実行時のメモリ所要量は，次に示す計算式で求めます。計算式の変数については (3) を参照してください。

### (1) HiRDB/ シングルサーバの場合

| 条件 | メモリ所要量の計算式（単位：キロバイト） |
|---|---|
| 32 ビットモードの場合 | 4352 ＋↑{（α ＋ γ）÷ 1024}↑＋ 1482 ＋↑{（β ＋ δ）÷ 1024}↑ |
| 64 ビットモードの場合 | 18050 ＋↑{（α ＋ γ）÷ 1024}↑＋ 1482 ＋↑{（β ＋ δ）÷ 1024}↑ |

### (2) HiRDB/ パラレルサーバの場合

| 条件 | | メモリ所要量の計算式（単位：キロバイト） |
|---|---|---|
| 32 ビットモードの場合 | MGR | 1530 ＋↑{α ÷ 1024}↑ |
| | 入力ファイルがあるサーバマシン | 1577 ＋↑{（β ＋ δ）÷ 1024}↑ |
| | BES ※ | 2305 ＋↑{γ ÷ 1024}↑ |
| 64 ビットモードの場合 | MGR | 4732 ＋↑{α ÷ 1024}↑ |
| | 入力ファイルがあるサーバマシン | 4663 ＋↑{（β ＋ δ）÷ 1024}↑ |
| | BES ※ | 12220 ＋↑{γ ÷ 1024}↑ |

注※

1 サーバマシン内に複数のバックエンドサーバがある場合は，バックエンドサーバの数だけ繰り返し計算（加算）してください。

### (3) 計算式で使用する変数

α（単位：バイト）：

{3056 ＋ A ＋ B ＋（516 × a）＋（572 × b）＋（312 × c）＋（144 × d）＋（8 × e）＋（1032 × f）＋（44 × g）＋（272 × h）＋（224 × i）＋（44 × j）＋（60 × k）＋（260 ×

m）＋（56 × n）＋（196 × p）＋（236 × q）＋（744 × r）＋（620 × s）} × 2

β（単位：バイト）：
{6908 ＋ α ＋（C × t）＋ K ＋（48 × a）＋（22 × b）＋（8 × e）＋（240 × i）＋（48 × j）＋（4 × k）＋（224 × m）＋（47416 × t）＋（1032 × u）＋（4 × v）}

γ（単位：バイト）：
{37700 ＋（α ÷ 2）＋ C ＋ D ＋ F ＋ H ＋ P ＋ Q ＋（80 × a）＋（1871 × b）＋（120 × c）＋（26 × g）＋（1532 × i）＋（36 × j）＋（44 × k）＋（1212 × m）＋（40 × n）＋（344 × p）＋（30 × q）＋（16 × u）＋（88 × v）＋（20 × w）}

δ（単位：バイト）：
{69436 ＋ α ＋ D ＋ K ＋ L ＋ M ＋ N ＋ S ＋（32 × a）＋（88 × c）＋（4 × g）＋（2156 × k）＋（24 × t）＋（1024 × u）＋（4 × v）＋（50 × y）＋（50 × z）}

a：列数

b：抽象データ型の列数

c：コンストラクタ又は逆コンストラクタ関数のパラメタ数

d：コマンドライン及び制御情報ファイルに指定したファイルパス名の数

e：LOB 中間ファイル指定数

f：列単位 LOB ファイル指定数

g：表格納 RD エリア数

h：表の横分割条件数

i：インデクス数

j：インデクス格納 RD エリア数

k：BLOB 型の列数

m：プラグインインデクス数

n：LOB 属性の抽象データ型を格納するユーザ LOB 用 RD エリア数

p：プラグインが提供する関数の数

q：プラグイン提供関数のパラメタ数

r：データ型プラグイン数

s：インデクス型プラグイン数

t：表格納サーバ数

u：使用するコンストラクタ関数のパラメタのうちで BLOB 型のパラメタ数

v：LOB 列を格納するユーザ LOB 用 RD エリア数

w：プラグインインデクスを格納するユーザ LOB 用 RD エリア数

y：BINARY 型の列数

skipdata 制御文で入力データ中の処理対象外とした BINARY データの列数と，実際に表に定義されている列数の合計です。

z：プラグイン提供関数の BINARY 属性のパラメタ数
skipdata 制御文で入力データ中の処理対象外としたプラグイン提供関数の BINARY データの列数と，実際に表に定義されている列数の合計です。

A：コマンドライン上で指定したファイルサイズの合計

B：コマンドライン及び制御情報ファイルに指定したファイルパス名の長さの合計

C：次に示す条件を満たす場合は（pd_utl_buff_size × 1024 ＋ 4096）× 2，そのほかの場合は 0
- HiRDB/ パラレルサーバの場合
  source 文に指定したサーバ名と表格納 RD エリアがあるバックエンドサーバ名が異なる pdload コマンド実行時，又は -g オプションを指定した pdrorg コマンド実行時

D：行長
表を構成する全列の定義長の合計です。ただし，BLOB 型や抽象データ型は 8 バイトとします。
BINARY 型は pdload の場合は定義長，pdrorg の場合は MIN（定義長，32500）バイトとします。また，FIX 表以外は（ａ ＋ 1）× 4 を加算します。

F：550 × 1024 ＋ sort 文に指定したワークバッファサイズ× 1024 ＋ G
-i オプションに c を指定した場合に加算します。

G：256（32 ビットモードの場合）又は 512（64 ビットモードの場合）

H：一括入出力用ローカルバッファの指定値× RD エリアのページ長× Ｊ ＋ランダムアクセス用ローカルバッファの指定値× RD エリアのページ長
-n オプションを指定した場合に加算します。RD エリアのページ長が横分割した RD エリアごとに異なる場合，最大のページ長で計算してください。

J：次に示す表から求めてください。

| -n オプションの指定値 | 表の分割種別 | | | | |
|---|---|---|---|---|---|
| | 非分割表 | キーレンジ分割表 | ハッシュ分割表 | | |
| | | | リバランスハッシュ（HASHA ～ HASHF） | | 非リバランスハッシュ（HASH0 ～ HASH6） |
| | | | FIX ハッシュ | フレキシブルハッシュ | |
| div 指定あり | 1 | 表のサーバ内横分割数 | HiRDB/ シングルサーバの場合：1024<br>HiRDB/ パラレルサーバの場合：（↑ 1024 ÷ g ↑）×（該当サーバ内の表格納 RD エリア数） | 表のサーバ内横分割数 | 表のサーバ内横分割数 |
| div 指定なし | | 1 | | | 1 |

K：パラメタ長
抽象データ型に格納する値を生成するコンストラクタ関数の引数の合計長です。ただし，BLOB 型のパラメタは 8 バイトで計算します。

L：次に示す条件をすべて満たす場合は 1 又は 2 の値を加算，そのほかの場合は 0
- source 文に errdata オペランドを指定
- source 文に指定したサーバ名と表格納 RD エリアがあるバックエンドサーバ名が異なる（HiRDB/パラレルサーバの場合）
- 表に抽象データ型の列がある又はユニークインデクスを定義している

1. errwork オペランド指定時：errwork オペランドの指定値× 1024
2. errwork オペランド省略時：pd_utl_buff_size × 1024 × 3 × t

M：UOC のメモリ所要量
UOC を使用した場合に加算します。

N：maxreclen オペランドを指定した場合に次に示す値を加算します。

入力データファイルが拡張 DAT 形式のとき：
maxreclen オペランド指定値× 1024

入力データファイルが DAT 形式のとき：
maxreclen オペランド指定値× 1024 × 3

対象となる表が BINARY 型の列を持ち，入力データファイルがバイナリ形式のとき：
次のどちらか小さい方を加算してください。
- maxreclen オペランド指定値× 1024
- 変数 D（行長）

上記以外のとき：
0

P：プラグインのメモリ所要量
プラグインが提供する抽象データ型の列がある場合に加算します。プラグインのメモリ所要量については，プラグインのマニュアルを参照してください。
コンストラクタ関数の引数が BLOB 型又は BINARY 型の場合は，1 行に定義されている（全抽象データ型に格納する実際のパラメタ長× 2）を加算します。

Q：出力バッファ用メモリ所要量
インデクス作成方法にインデクス一括作成モード，又はインデクス情報出力モードを指定していて，次の条件を満たす場合，2 メガバイトを加算します。
- 表分割数×インデクス定義数＞プロセスのオープン数の上限 -576

S：(4 ×バッファ長＋ 1.1）× 1024
バッファ長は次の値になります（値は 32 キロバイト単位に切り上げてください)。
- データベース作成ユティリティ（pdload）の option 文の file_buff_size の指定値
- 上記の指定がない場合は，システム共通定義の pd_utl_file_buff_size の指定値
- 上記のどちらの指定もない場合は 32

## 19.2.4 データベース再編成ユティリティ（pdrorg）実行時のメモリ所要量

データベース再編成ユティリティ（pdrorg）実行時のメモリ所要量は，次に示す計算式で求めます。計算式の変数については，「(3) 計算式で使用する変数」を参照してください。

## (1) HiRDB/シングルサーバの場合

| 条件 | メモリ所要量の計算式（単位：キロバイト） |
|---|---|
| 32 ビットモードの場合 | 5466 ＋↑｛（α ＋ γ）÷ 1024｝↑＋ 1621 ＋↑｛（β ＋ δ）÷ 1024｝↑ |
| 64 ビットモードの場合 | 18362 ＋↑｛（α ＋ γ）÷ 1024｝↑＋ 1621 ＋↑｛（β ＋ δ）÷ 1024｝↑ |

## (2) HiRDB/パラレルサーバの場合

| 条件 | | | メモリ所要量の計算式（単位：キロバイト） |
|---|---|---|---|
| 32 ビットモードの場合 | MGR | | 1509 ＋↑｛α ÷ 1024｝↑ |
| | -g オプションなし | DS | 1413 ＋↑｛β ÷ 1024｝↑ |
| | | BES※1※2 | 3505 ＋↑｛（γ ＋ δ）÷ 1024｝↑ |
| | -g オプションあり | アンロードデータファイルがあるサーバマシン | 1413 ＋↑｛（β ＋ δ）÷ 1024｝↑ |
| | | BES※2 | 3505 ＋↑｛γ ÷ 1024｝↑ |
| 64 ビットモードの場合 | MGR | | 4761 ＋↑｛β ÷ 1024｝↑ |
| | -g オプションなし | DS | 16482 ＋↑｛β ÷ 1024｝↑ |
| | | BES※1※2 | 12100 ＋↑｛（γ ＋ δ）÷ 1024｝↑ |
| | -g オプションあり | アンロードデータファイルがあるサーバマシン | 4566 ＋↑｛（β ＋ δ）÷ 1024｝↑ |
| | | BES※2 | 12100 ＋↑｛γ ÷ 1024｝↑ |

注※1

ディクショナリ表を再編成する場合は，計算値をディクショナリサーバがあるサーバマシンに加算してください。

注※2

1 サーバマシン内に複数のバックエンドサーバがある場合は，バックエンドサーバの数だけ繰り返し計算（加算）してください。

## (3) 計算式で使用する変数

α（単位：バイト）：

｛2592 ＋ A ＋ B ＋（116 × a）＋（260 × b）＋（6 × c）＋（272 × d）＋（44 × g）＋（272 × h）＋（224 × i）＋（44 × j）＋（60 × k）＋（260 × m）＋（56 × n）＋（196 × p）＋（236 × q）＋（744 × r）＋（620 × s）＋（24 × t）｝× 2

β（単位：バイト）：

｛40940 ＋ α ＋（C × t）＋（D × t）＋（136 × a）＋（56 × g）＋（2200 × j）＋（4 × k）＋（548 × t）｝

γ（単位：バイト）：

｛101140 ＋（α ÷ 2）＋ C ＋ D ＋ F ＋ H ＋ P ＋ Q ＋（128 × a）＋（1949 × b）＋（120 × c）＋（154 × g）＋（336 × i）＋（216 × j）＋（32056 × k）＋（1212 × m）＋（131224 × n）＋（344 × p）＋（30 × q）＋（20 × w）｝

δ（単位：バイト）：
{33104 ＋ α ＋（K × 2）＋ S ＋（204 × a）＋（688 × b）＋（306 × c）＋（44 × g）＋（272 × h）＋（224 × i）＋（44 × j）＋（56 × n）＋（716 × r）＋（152 × v）}

注

- 変数については，「19.2.3　データベース作成ユティリティ（pdload）実行時のメモリ所要量」の「(3) 計算式で使用する変数」で説明しています。
- ディクショナリ再編成又はスキーマ単位の再編成で 1 回のコマンドで複数の表を処理する場合，変数 a ～ z は処理対象すべての表の合計で計算してください。

## 19.2.5　データベース構成変更ユティリティ（pdmod）実行時のメモリ所要量

データベース構成変更ユティリティ（pdmod）実行時のメモリ所要量は，次に示す計算式で求めます。

### (1) HiRDB/ シングルサーバの場合

| 条件 | メモリ所要量の計算式（単位：キロバイト） |
|---|---|
| 32 ビットモードの場合 | ↑{<br>4 × a ＋ 56016 × b ＋ 53016 × c ＋ 2440 × d ＋ 1724 × e<br>＋（94008 × f）÷ 500 ＋（4008 × g）÷ 1000 ＋ 441231 ＋ h ＋ i ＋ j ＋ k<br>}÷ 1024 ↑＋ 9.8 |
| 64 ビットモードの場合 | ↑{<br>4 × a ＋ 56024 × b ＋ 53024 × c ＋ 3040 × d ＋ 1736 × e<br>＋（100016 × f）÷ 500 ＋（4012 × g）÷ 1000 ＋ 451231 ＋ h ＋ i ＋ j ＋ k<br>}÷ 1024 ↑＋ 9.8 |

a：pd_max_rdarea_no の値

b：initialize rdarea 文実行時の自 RD エリア内のインデクス数＋他 RD エリアのインデクス数

c：initialize rdarea 文実行時の LOB 列の総数

d：initialize rdarea 文実行時の LOB 属性の抽象データ型の総数

e：initialize rdarea 文実行時のプラグイン列とプラグインインデクスの総数

f：initialize rdarea 文実行時の抽象データ型の総数

g：initialize rdarea 文実行時の自 RD エリア格納表の ASSIGN LIST の総数

h：8 × a ＋ 30720
alter HiRDB mode to parallel 文で HiRDB/ シングルサーバから HiRDB/ パラレルサーバへ移行する場合に加算します。

i：46744
create rdarea 文でデータディクショナリ LOB 用 RD エリアを追加する場合に加算します。

j：88064
alter system 文でディクショナリ表の参照権限を変更する場合に加算します。

k：54732
alter system 文でディクショナリ表の列属性 MCHAR にする場合に加算します。

## (2) HiRDB/パラレルサーバの場合

| 条件 | | メモリ所要量の計算式（単位：キロバイト） |
|---|---|---|
| 32 ビットモードの場合 | DS | ↑ {<br>4 × a ＋ 56016 × b ＋ 53016 × c ＋ 2440 × d ＋ 1724 × e<br>＋（94008 × f）÷ 500 ＋（4008 × g）÷ 1000 ＋ 441231 ＋ h ＋ i ＋ j<br>＋ 108428 × m<br>} ÷ 1024 ↑ |
| | BES | ↑（4 × a ＋ 253266 ＋ k）÷ 1024 ↑ |
| | FES | 0.52 |
| | MGR | 9.8 |
| 64 ビットモードの場合 | DS | ↑ {<br>4 × a ＋ 56024 × b ＋ 53024 × c ＋ 3040 × d ＋ 1736 × e<br>＋（100016 × f）÷ 500 ＋（4012 × g）÷ 1000 ＋ 451231 ＋ h ＋ i ＋ j<br>＋ 108432 × m<br>} ÷ 1024 ↑ |
| | BES | ↑（4 × a ＋ 261112 ＋ k）÷ 1024 ↑ |
| | FES | 0.53 |
| | MGR | 9.8 |

a：pd_max_rdarea_no の値

b：initialize rdarea 文実行時の自 RD エリア内のインデクス数＋他 RD エリアのインデクス数

c：initialize rdarea 文実行時の LOB 列の総数

d：initialize rdarea 文実行時の LOB 属性の抽象データ型の総数

e：initialize rdarea 文実行時のプラグイン列とプラグインインデクスの総数

f：initialize rdarea 文実行時の抽象データ型の総数

g：initialize rdarea 文実行時の自 RD エリア格納表の ASSIGN LIST の総数

h：46744

create rdarea 文でデータディクショナリ LOB 用 RD エリアを追加する場合に加算します。

i：88064

alter system 文でディクショナリ表の参照権限を変更する場合に加算します。

j：54732

alter system 文でディクショナリ表の列属性 MCHAR にする場合に加算します。

k：1600

initialize rdarea 文を実行する場合に加算します。

m：move rdarea 文を実行する場合に次に示す計算式を加算します。

move rdarea 文を実行しない場合 0 になります。

↑（192 ×移動対象 RD エリア数＋ 160 ×移動対象 RD エリアの総 HiRDB ファイル数＋ 8 ×移動対象 RD エリアに格納されている表の総数＋ 8 ×移動対象 RD エリアに格納されているインデクスの総数＋ 8 ×移動対象 RD エリアに格納されている LOB 列の総数）÷ 102400 ↑

## 19.2.6 統計解析ユティリティ（pdstedit）実行時のメモリ所要量

統計解析ユティリティ（pdstedit）実行時のメモリ所要量は，次に示す計算式で求めます。

| 条件 | メモリ所要量の計算式（単位：キロバイト） |
|---|---|
| HiRDB/ シングルサーバの場合 | 16384 ※＋ 1.5 ×ホスト数× HiRDB サーバ数× UAP 数 |
| HiRDB/ パラレルサーバの場合 | |

注※

64 ビットモードの場合は 18640 です。

## 19.2.7 データベース状態解析ユティリティ（pddbst）実行時のメモリ所要量

データベース状態解析ユティリティ（pddbst）実行時のメモリ所要量は，次に示す計算式で求めます。

| 条件 | | メモリ所要量の計算式（単位：キロバイト） |
|---|---|---|
| HiRDB/ シングルサーバの場合（32 ビットモードの場合） | | RD エリア物理解析時：4600<br>RD エリア論理解析時：4400 ＋ 1.4 × a ＋↑ a ÷ 100 ↑× 672<br>表単位の解析時：4400 ＋ 1.4 × b ＋↑ b ÷ 100 ↑× 672<br>インデクス単位の解析時：4400 ＋ 1.4 × c ＋↑ c ÷ 100 ↑× 672<br>クラスタキーの解析時：4600 |
| HiRDB/ シングルサーバの場合（64 ビットモードの場合） | | RD エリア物理解析時：17915<br>RD エリア論理解析時：17915 ＋ 1.4 × a ＋↑ a ÷ 100 ↑× 672<br>表単位の解析時：17915 ＋ 1.4 × b ＋↑ b ÷ 100 ↑× 672<br>インデクス単位の解析時：17915 ＋ 1.4 × c ＋↑ c ÷ 100 ↑× 672<br>クラスタキーの解析時：17915 |
| HiRDB/ パラレルサーバの場合（32 ビットモードの場合） | DS | RD エリア物理解析時：750<br>RD エリア論理解析時：750 ＋ 0.1 × a<br>表単位の解析時：750 ＋ 0.1 × b<br>インデクス単位の解析時：750 ＋ 0.1 × c<br>クラスタキーの解析時：750 |
| | BES | RD エリア物理解析時：650<br>RD エリア論理解析時：650 ＋ 0.1 × a<br>表単位の解析時：650 ＋ 0.1 × b<br>インデクス単位の解析時：650 ＋ 0.1 × c<br>クラスタキーの解析時：650 |
| | MGR | RD エリア物理解析時：3900<br>RD エリア論理解析時：3550 ＋ 1.3 × a ＋↑ a ÷ 100 ↑× 672<br>表単位の解析時：3550 ＋ 1.3 × b ＋↑ b ÷ 100 ↑× 672<br>インデクス単位の解析時：3550 ＋ 1.3 × c ＋↑ c ÷ 100 ↑× 672<br>クラスタキーの解析時：3900 |
| HiRDB/ パラレルサーバの場合（64 ビットモードの場合） | DS | RD エリア物理解析時：16312<br>RD エリア論理解析時：16312 ＋ 0.1 × a<br>表単位の解析時：16312 ＋ 0.1 × b<br>インデクス単位の解析時：16312 ＋ 0.1 × c<br>クラスタキーの解析時：16312 |

| 条件 | | メモリ所要量の計算式（単位：キロバイト） |
|---|---|---|
| | BES | RD エリア物理解析時：11812<br>RD エリア論理解析時：11812 ＋ 0.1 × a<br>表単位の解析時：11812 ＋ 0.1 × b<br>インデクス単位の解析時：11812 ＋ 0.1 × c<br>クラスタキーの解析時：11812 |
| | MGR | RD エリア物理解析時：4601<br>RD エリア論理解析時：4601 ＋ 1.3 × a ＋↑ a ÷ 100 ↑× 672<br>表単位の解析時：4601 ＋ 1.3 × b ＋↑ b ÷ 100 ↑× 672<br>インデクス単位の解析時：4601 ＋ 1.3 × c ＋↑ c ÷ 100 ↑× 672<br>クラスタキーの解析時：4601 |

a：RD エリア格納表数＋ RD エリア格納インデクス数

b：表格納 RD エリア数

c：インデクス格納 RD エリア数

## 19.2.8　最適化情報収集ユティリティ（pdgetcst）実行時のメモリ所要量

最適化情報収集ユティリティ（pdgetcst）実行時のメモリ所要量は，次に示す計算式で求めます。

| 条件 | | メモリ所要量の計算式（単位：キロバイト） |
|---|---|---|
| HiRDB/ シングルサーバの場合 | | 11763 ＋ 0.1 ×表格納 RD エリア数＋ 0.07 ×インデクス格納 RD エリア数＋ 1.0 ×インデクス数＋ 0.04 ×スキーマ内表数＋ 1.0 ×サーバ数＋ 11 ×列数 |
| HiRDB/ パラレルサーバの場合 | BES | 11772 ＋ 0.06 ×表格納 RD エリア数＋ 0.05 ×インデクス格納 RD エリア数＋ 0.03 ×インデクス数 |
| | MGR | 16402 |
| | DS | 4384 ＋ 0.1 ×表格納 RD エリア数＋ 0.07 ×インデクス格納 RD エリア数＋ 1.0 ×インデクス数＋ 0.04 ×スキーマ内表数＋ 1.0 ×サーバ数＋ 11 ×列数 |

注

上記のほかに次の SQL が使用するメモリ所要量を加算します。

```
SELECT 内部情報※,インデクス第一構成列名 FROM 認可識別子.表識別子
    ORDER BY インデクス第一構成列名 WITHOUT LOCK NOWAIT;
SELECT FLOAT (COUNT(*)) FROM 認可識別子.表識別子 WITHOUT LOCK NOWAIT;
SELECT FLOAT (COUNT(インデクス第一構成列名))
     FROM 認可識別子.表識別子 WITHOUT LOCK NOWAIT;
ALTER TABLE 認可識別子.表識別子 CHANGE LOCK ROW;
```

注※

内部情報は 12 バイトです。したがって，インデクス第一構成列のほかに 12 バイトの文字列（CHAR(12)）を検索する SQL を発行したものとして見積もってください。

SQL が使用するメモリ所要量については，次の箇所を参照してください。

- HiRDB/ シングルサーバの場合

「15.1.6　SQL 実行時に必要なメモリ所要量の計算式」

「15.1.7　SQL 前処理時に必要なメモリ所要量の計算式」

- HiRDB/ パラレルサーバの場合

「15.2.6　SQL 実行時に必要なメモリ所要量の計算式」

「15.2.7　SQL 前処理時に必要なメモリ所要量の計算式」

## 19.2.9 データベース複写ユティリティ（pdcopy）実行時のメモリ所要量

データベース複写ユティリティ（pdcopy）実行時のメモリ所要量は，次に示す計算式で求めます。

### （1） HiRDB/シングルサーバの場合

| 条件 | メモリ所要量の計算式（単位：キロバイト） |
|---|---|
| シングルサーバ | 88 ＋バックアップファイル数× 2 × MAX（32，pd_utl_buff_size の値）<br>＋バックアップファイル数×｛(バックアップ対象 RD エリア数＋ 9）÷ 10｝× 6<br>＋｛(全バックアップ対象 RD エリア構成ファイル数＋ 25）÷ 16｝× 8 ＋ 100<br>＋ 49 ＋バックアップファイル数× 2 × MAX（32，pd_utl_buff_size の値）＋ 64<br>＋バックアップファイル数×｛(バックアップ対象 RD エリア数＋ 9）÷ 10｝× 6<br>＋｛(全バックアップ対象 RD エリア構成ファイル数＋ 25）÷ 16｝× 8 ＋ 100<br>**●バックアップファイルがこのサーバマシンにある場合に加算**<br>＋ 63 ＋バックアップファイル数×（2 × MAX（32，pd_utl_buff_size の値）× 2<br>＋バックアップファイル数×｛(バックアップ対象 RD エリア数＋ 9）÷ 10｝× 6<br>＋｛(全バックアップ対象 RD エリア構成ファイル数＋ 25）÷ 16｝× 8 ＋ 100 ＋ c<br>**●差分バックアップを取得する場合に加算**<br>＋ 32 × 2 ＋↑（512 ＋ 128 ×バックアップ対象 RD エリア数）÷ 32768 ↑× 32<br>＋↑（256 ＋ 128 ×バックアップ対象 RD エリア数＋ a ＋ 8 × b）÷ 32768 ↑× 32 × 2 |

a：-b オプションに指定するバックアップファイルの名称長（バイト）
複数のバックアップファイルを指定する場合は，合計の名称長になります。

b：-b オプションに指定するバックアップファイルの数

c：pd_utl_file_buff_size の指定値（値は 32 キロバイト単位に切り上げ）
pd_utl_file_buff_size の指定がない場合は 32

### （2） HiRDB/パラレルサーバの場合

| 条件 | メモリ所要量の計算式（単位：キロバイト） |
|---|---|
| MGR | 88 ＋バックアップファイル数× 2 × MAX（32，pd_utl_buff_size の値）<br>＋バックアップファイル数×｛(バックアップ対象 RD エリア数＋ 9）÷ 10｝× 6<br>＋｛(全バックアップ対象 RD エリア構成ファイル数＋ 25）÷ 16｝× 8 ＋ 100<br>**●差分バックアップを取得する場合に加算**<br>＋ 32 × 2 ＋↑（512 ＋ 128 ×バックアップ対象 RD エリア数）÷ 32768 ↑× 32<br>＋↑（256 ＋ 128 ×バックアップ対象 RD エリア数＋ a ＋ 8 × b）÷ 32768 ↑× 32 × 2 |
| DS<br>BES | 49 ＋バックアップファイル数× 2 × MAX（32，pd_utl_buff_size の値）＋ 64<br>＋バックアップファイル数×｛(バックアップ対象 RD エリア数＋ 9）÷ 10｝× 6<br>＋｛(全バックアップ対象 RD エリア構成ファイル数＋ 25）÷ 16｝× 8 ＋ 100 |
| バックアップファイルがあるサーバマシン | 63 ＋バックアップファイル数× 2 × MAX（32，pd_utl_buff_size の値）<br>×(バックアップ対象サーバ数＋ 1)<br>＋バックアップファイル数×｛(バックアップ対象 RD エリア数＋ 9）÷ 10｝× 6<br>＋｛(全バックアップ対象 RD エリア構成ファイル数＋ 25）÷ 16｝× 8 ＋ 100 ＋ c |

a：-b オプションに指定するバックアップファイルの名称長（バイト）
複数のバックアップファイルを指定する場合は，合計の名称長になります。

b：-b オプションに指定するバックアップファイルの数

c：pd_utl_file_buff_size の指定値（値は 32 キロバイト単位に切り上げ）
pd_utl_file_buff_size の指定がない場合は 32

## 19.2.10　データベース回復ユティリティ（pdrstr）実行時のメモリ所要量

データベース回復ユティリティ（pdrstr）実行時のメモリ所要量は，次に示す計算式で求めます。

### （1）HiRDB/シングルサーバの場合

| 条件 | メモリ所要量の計算式（単位：キロバイト） |
|---|---|
| シングルサーバ | 65 ＋｛(回復対象 RD エリア数＋ 9）÷ 10｝× 6<br>＋｛(回復対象 RD エリアの構成ファイル数＋ 25）÷ 16｝× 8 ＋ 50<br>＋ 98 ＋ 2 × MAX（32，pd_utl_buff_size の値）<br>＋｛(回復対象 RD エリア数＋ 9）÷ 10｝× 6 ＋ c<br>＋｛(回復対象 RD エリアの構成ファイル数＋ 25）÷ 16｝× 8 ＋ 100<br>＋｛(回復対象 RD エリア数＋ 99）÷ 100｝× 5<br>**●バックアップファイルがこのサーバマシンにある場合に加算**<br>＋ 100 ＋ 2 × MAX（32，pd_utl_buff_size の値）<br>＋｛(回復対象 RD エリア数＋ 9）÷ 10｝× 6<br>＋｛(回復対象 RD エリアの構成ファイル数＋ 25）÷ 16｝× 8 ＋ 100 ＋ e<br>**●アンロードログファイル，又はシステムログファイルを入力する場合に加算**<br>＋ 57 ＋ 2 × MAX（32，pd_utl_buff_size の値）<br>＋｛(回復対象 RD エリア数＋ 9）÷ 10｝× 6 ＋ 64<br>＋｛(回復対象 RD エリアの構成ファイル数＋ 25）÷ 16｝× 8 ＋ 100 ＋ d<br>＋ 0.6 ×回復対象 RD エリア数＋ソート用ワークバッファサイズ（-y オプションの値）<br>**●差分バックアップを使用した回復をする場合に加算**<br>＋ 32 × 2 ＋↑（512 ＋ 128 ×バックアップ対象 RD エリア数）÷ 32768 ↑× 32<br>＋↑（256 ＋ 128 ×バックアップ対象 RD エリア数＋ a ＋ 8 × b）÷ 32768 ↑× 32<br>＋↑（32 ×差分バックアップの継続回数）÷ 1024 ↑ |

a：-b オプションに指定するバックアップファイルの名称長（バイト）
　複数のバックアップファイルを指定する場合は，合計の名称長になります。

b：-b オプションに指定するバックアップファイルの数

c：書き込みバッファサイズを指定している場合は MAX（64，書き込みバッファサイズ），指定していない場合は 60
　書き込みバッファサイズは -Y オプションの指定値になります。

d：

- 32 ビットモードの場合：
  640 ＋ 8 ×↑最大同時実行トランザクション数÷ 100 ↑
  ＋ 5 ×↑回復対象 RD エリア数÷ 100 ↑
  ＋回復対象 RD エリアの最大ページサイズ× 54
  ＋ 9 ×ロールバック対象トランザクション数
  ＋ 0.02 ×回復対象 RD エリアの構成ファイル数
  ＋↑（304 ＋ 36 ＋ 4 ×（回復対象 RD エリア数－ 1）
  　＋ 352 ＋ 304 ×（回復対象 RD エリア数－ 1）
  　＋ 96 ＋ 4 ×（回復対象 RD エリア数－ 1）
  　＋ 384 ＋ 320 ×（回復対象 RD エリア数－ 1）＋ 16）÷ 1024 ↑
- 64 ビットモードの場合：
  640 ＋ 11 ×↑最大同時実行トランザクション数÷ 100 ↑
  ＋ 6 ×↑回復対象 RD エリア数÷ 100 ↑
  ＋回復対象 RD エリアの最大ページサイズ× 54
  ＋ 9 ×ロールバック対象トランザクション数

＋ 0.03 ×回復対象 RD エリアの構成ファイル数
＋↑（304 ＋ 40 ＋ 8 ×（回復対象 RD エリア数− 1）
＋ 400 ＋ 336 ×（回復対象 RD エリア数− 1）
＋ 168 ＋ 8 ×（回復対象 RD エリア数− 1）
＋ 408 ＋ 336 ×（回復対象 RD エリア数− 1）＋ 16）÷ 1024 ↑

e：pd_utl_file_buff_size の指定値（値は 32 キロバイト単位に切り上げ）
pd_utl_file_buff_size の指定がない場合は 32

## (2) HiRDB/ パラレルサーバの場合

| 条件 | メモリ所要量の計算式（単位：キロバイト） |
|---|---|
| MGR | 65 ＋ {(回復対象 RD エリア数＋ 9）÷ 10} × 6<br>＋ {(回復対象 RD エリアの構成ファイル数＋ 25）÷ 16} × 8 ＋ 50<br>**●差分バックアップを使用した回復をする場合に加算**<br>＋ 32 × 2 ＋↑（512 ＋ 128 ×バックアップ対象 RD エリア数）÷ 32768 ↑× 32<br>＋↑（256 ＋ 128 ×バックアップ対象 RD エリア数＋ a ＋ 8 × b）÷ 32768 ↑× 32<br>＋↑（32 ×差分バックアップの継続回数）÷ 1024 ↑ |
| DS | 35 ＋ 2 × MAX（32，pd_utl_buff_size の値）＋ 100<br>＋ 98 ＋ 2 × MAX（32，pd_utl_buff_size の値）<br>＋ {(回復対象 RD エリア数＋ 9）÷ 10} × 6 ＋ c<br>＋ {(回復対象 RD エリアの構成ファイル数＋ 25）÷ 16} × 8 ＋ 100<br>＋ {(回復対象 RD エリア数＋ 99）÷ 100} × 5<br>**●アンロードログファイル，又はシステムログファイルを入力する場合に加算**<br>＋ 57 ＋ 2 × MAX（32，pd_utl_buff_size の値）<br>＋ {(回復対象 RD エリア数＋ 9）÷ 10} × 6 ＋ 64<br>＋ {(回復対象 RD エリアの構成ファイル数＋ 25）÷ 16} × 8 ＋ 100 ＋ d<br>＋ 0.6 ×回復対象 RD エリア数＋ソート用ワークバッファサイズ（-y オプションの値） |
| BES | 98 ＋ 2 × MAX（32，pd_utl_buff_size の値）<br>＋ {(回復対象 RD エリア数＋ 9）÷ 10} × 6 ＋ c<br>＋ {(回復対象 RD エリアの構成ファイル数＋ 25）÷ 16} × 8 ＋ 100<br>＋ {(回復対象 RD エリア数＋ 99）÷ 100} × 5<br>**●アンロードログファイル，又はシステムログファイルを入力する場合に加算**<br>＋ 57 ＋ 2 × MAX（32，pd_utl_buff_size の値）<br>＋ {(回復対象 RD エリア数＋ 9）÷ 10} × 6 ＋ 64<br>＋ {(回復対象 RD エリアの構成ファイル数＋ 25）÷ 16} × 8 ＋ 100 ＋ d<br>＋ 0.6 ×回復対象 RD エリア数＋ソート用ワークバッファサイズ (-y オプションの値) |
| バックアップファイルがあるサーバマシン | 100 ＋ 2 × MAX（32，pd_utl_buff_size）×回復対象サーバ数<br>＋ {(回復対象 RD エリア数＋ 9）÷ 10} × 6<br>＋ {(回復対象 RD エリア構成ファイル数＋ 25）÷ 16} × 8 ＋ 100 ＋ e |

a：-b オプションに指定するバックアップファイルの名称長（バイト）
複数のバックアップファイルを指定する場合は，合計の名称長になります。

b：-b オプションに指定するバックアップファイルの数

c：書き込みバッファサイズを指定している場合は MAX（64，書き込みバッファサイズ），指定していない場合は 60
書き込みバッファサイズは -Y オプションの指定値になります。

d：
- 32 ビットモードの場合：
  640 ＋ 8 ×↑最大同時実行トランザクション数÷ 100 ↑

＋ 5 ×↑回復対象 RD エリア数÷ 100 ↑
＋回復対象 RD エリアの最大ページサイズ× 54
＋ 9 ×ロールバック対象トランザクション数
＋ 0.02 ×回復対象 RD エリアの構成ファイル数
＋↑（304 ＋ 36 ＋ 4 ×（回復対象 RD エリア数− 1）
＋ 352 ＋ 304 ×（回復対象 RD エリア数− 1）
＋ 96 ＋ 4 ×（回復対象 RD エリア数− 1）
＋ 384 ＋ 320 ×（回復対象 RD エリア数− 1）
＋ 16 ＋ 32 ×回復対象 RD エリア数）÷ 1024 ↑

- 64 ビットモードの場合：
  640 ＋ 11 ×↑最大同時実行トランザクション数÷ 100 ↑
  ＋ 6 ×↑回復対象 RD エリア数÷ 100 ↑
  ＋回復対象 RD エリアの最大ページサイズ× 54
  ＋ 9 ×ロールバック対象トランザクション数
  ＋ 0.03 ×回復対象 RD エリアの構成ファイル数
  ＋↑（304 ＋ 40 ＋ 8 ×（回復対象 RD エリア数− 1）
  ＋ 400 ＋ 336 ×（回復対象 RD エリア数− 1）
  ＋ 168 ＋ 8 ×（回復対象 RD エリア数− 1）
  ＋ 408 ＋ 336 ×（回復対象 RD エリア数− 1）
  ＋ 16 ＋ 48 ×回復対象 RD エリア数）÷ 1024 ↑

e：pd_utl_file_buff_size の指定値（値は 32 キロバイト単位に切り上げ）
pd_utl_file_buff_size の指定がない場合は 32

## 19.2.11 ディクショナリ搬出入ユティリティ（pdexp）実行時のメモリ所要量

ディクショナリ搬出入ユティリティ（pdexp）実行時のメモリ所要量は，次に示す計算式で求めます。

### (1) HiRDB/ シングルサーバの場合

| 条件 | | メモリ所要量の計算式（単位：キロバイト） |
|---|---|---|
| 32 ビットモード | SDS | 1307 ＋ 6.7 ×↑ CTL ÷ 100 ↑＋ 0.07 × CTL |
| | 搬出ファイルのあるホスト | 4499 ＋ 0.07 × CTL ＋ 0.11 × CHK ＋ 1.5 × FKY ＋ CSZ ＋ 0.5 × CMN ＋ 0.01 ×（CHK ＋ FKY ＋ CMN ＋ DIV ＋ 10）＋ 0.6 × DIV ＋ DEF ＋ 0.06 × LOB ＋ 0.2 × TBL ＋ SQL |
| 64 ビットモード | SDS | 1494 ＋ 6.7 ×↑ CTL ÷ 100 ↑＋ 0.07 × CTL |
| | 搬出ファイルのあるホスト | 4582 ＋ 0.07 × CTL ＋ 0.11 × CHK ＋ 1.5 × FKY ＋ CSZ ＋ 0.5 × CMN ＋ 0.01 ×（CHK ＋ FKY ＋ CMN ＋ DIV ＋ 10）＋ 0.6 × DIV ＋ DEF ＋ 0.06 × LOB ＋ 0.2 × TBL ＋ SQL |

### (2) HiRDB/ パラレルサーバの場合

| 条件 | | メモリ所要量の計算式（単位：キロバイト） |
|---|---|---|
| 32 ビットモード | MGR | 1714 ＋ 6.7 ×↑ CTL ÷ 100 ↑＋ 0.07 × CTL |
| | 搬出ファイルのあるホスト | 4907 ＋ 0.07 × CTL ＋ 0.11 × CHK ＋ 1.5 × FKY ＋ CSZ ＋ 0.5 × CMN ＋ 0.01 ×（CHK ＋ FKY ＋ CMN ＋ DIV ＋ 10）＋ 0.6 × DIV ＋ DEF ＋ 0.06 × LOB ＋ 0.2 × TBL ＋ SQL |
| 64 ビットモード | MGR | 2016 ＋ 6.7 ×↑ CTL ÷ 100 ↑＋ 0.07 × CTL |
| | 搬出ファイルのあるホスト | 5293 ＋ 0.07 × CTL ＋ 0.11 × CHK ＋ 1.5 × FKY ＋ CSZ ＋ 0.5 × CMN ＋ 0.01 ×（CHK ＋ FKY ＋ CMN ＋ DIV ＋ 10）＋ 0.6 × DIV ＋ DEF ＋ 0.06 × LOB ＋ 0.2 × TBL ＋ SQL |

### (3) 計算式で使用する変数

CMN：列数

CHK：検査制約数

CSZ：検査制約の検索条件の合計サイズ（SQL_TABLES 表の CHK_SOURCE_LEN の値）（単位：バイト）

CTL：制御文ファイルに指定された制御数

DEF：DEFAULT 句の最大定義長（単位：バイト）

DIV：分割条件数

FKY：外部キーの数

LOB：LOB 格納用 ＲＤ エリア数

SQL：次の SQL 文が使用するメモリ所要量

- 表の搬出入の場合：CREATE TABLE 文
- プロシジャの搬出入の場合：CREATE PROCEDURE 文
- トリガの搬出入の場合：CREATE TRIGGER 文

これらのメモリ所要量については，「15.1.6　SQL 実行時に必要なメモリ所要量の計算式」，及び「15.1.7　SQL 前処理時に必要なメモリ所要量の計算式」を参照してください。

TBL：実際に搬出入する表，プロシジャ，トリガ数（プロシジャ，トリガの場合は CTL と同じ）

## 19.2.12 アクセスパス表示ユティリティ（pdvwopt）実行時のメモリ所要量

アクセスパス表示ユティリティ（pdvwopt）実行時のメモリ所要量は，次に示す計算式で求めます。

| 条件 | | メモリ所要量の計算式（単位：キロバイト） |
|---|---|---|
| HiRDB/ シングルサーバの場合 | | $\sum_{i=1}^{a} b_i \times 0.7 + 200$ |
| HiRDB/ パラレルサーバの場合 | FES | |

a：SQL 中の問合せ数

$b_i$：問合せ中の表数

## 19.2.13　リバランスユティリティ（pdrbal）実行時のメモリ所要量

リバランスユティリティ（pdrbal）実行時のメモリ所要量は，次に示す計算式で求めます。

### (1) HiRDB/ シングルサーバの場合

**メモリ所要量の計算式（単位：キロバイト）**

8756※3＋536＋0.02 ×列数＋0.2 ×移動先 RD エリア数
＋1.7 ×移動元 RD エリア数＋0.26 ×移動先 RD エリア数×インデクス数
＋（0.09 ＋↑平均 index 文ファイル長÷ 1024 ↑）× index 文数
＋（0.02 ＋↑平均ディレクトリ長÷ 1024 ↑）×（idxwork 文数＋ sort 文数）
＋↑（制御情報ファイル長＋実行結果ファイル長）÷ 1024 ↑
＋0.05 ×列数＋0.05 × RD エリア数＋0.15 ×インデクス数
＋0.05 ×インデクス格納 RD エリア数
＋550 × 1024 ＋ソートワークサイズ※1

**●-n オプションを指定した場合**
＋ RD エリアのページ長※2 ×一括入出力バッファ面数× y
**●対象表に LOB 列がある場合**
＋64 ＋0.01 × LOB 列数＋0.18 ×移動先 RD エリア数
＋0.09 ×移動元 RD エリア数＋0.08 × LOB 格納 RD エリア数
**●対象表に BINARY 列がある場合**
＋33 ×（BINARY 列数×移動先 RD エリア数×移動元 RD エリア数）

**●対象表にプラグインが提供する抽象データ型がある場合**
＋40 ＋（0.27 ＋2 ×抽象データ型長）×抽象データ型の列数
＋0.3 × unld_func 指定数
＋（128 ＋0.11 × LOB 属性数＋0.1 × unld_func 指定関数数
＋0.07 ×抽象データ型属性数）×抽象データ型の列数
＋（33 × BINARY 属性数×移動先 RD エリア数×移動元 RD エリア数）× 2
＋0.01 ×プラグインインデクス数＋0.19 × unld_func 文数
＋（↑平均 unld_func 文長÷ 1024 ↑× unld_func 文数）
＋（↑平均 reld_func 文長÷ 1024 ↑× reld_func 文数）
＋抽象データ型列数× 1 ＋（LOB 属性数× 0.05）× RD エリア数
＋データ型プラグイン数× 10 ＋プラグインインデクス数× 10
＋プラグインのメモリ所要量

**●インデクス作成方法に，インデクス一括作成モード又はインデクス情報出力モードを指定していて，次の条件を満たす場合**
**・表分割数×インデクス定義数＞プロセスのオープン数の上限** -576
＋2048

y：次に示すどちらかの値を代入してください。

- FIX ハッシュ分割表にリバランス機能を使用した場合
  （↑ 1024 ÷表全体の格納 RD エリア数↑）×該当サーバ内の表格納 RD エリア数
- そのほかの場合
  1

注※1
インデクスの一括作成時（-i c 又は省略）に加算します。

注※2
表を横分割した RD エリアごとにページ長が異なる場合は，ページ長の最大値で計算してください。

注※3

64 ビットモードの場合は 14272 です。

## (2) HiRDB/パラレルサーバの場合

| 条件 | メモリ所要量の計算式（単位：キロバイト） |
|---|---|
| MGR | 1498※3＋ 2 ＋ 0.05 ×列数＋ 0.05 × RD エリア数＋ 0.15 ×インデクス数<br>＋ 0.05 ×インデクス格納 RD エリア数<br>＋（0.09 ＋↑平均 index 文ファイル長÷ 1024 ↑）× index 文数<br>＋（0.02 ＋↑平均ディレクトリ長÷ 1024 ↑）×（idxwork 文数＋ sort 文数）<br>＋↑（制御情報ファイル長＋実行結果ファイル長）÷ 1024 ↑<br>**●対象表に LOB 列がある場合**<br>＋ 0.08 × LOB 格納 RD エリア数<br>**●対象表にプラグインが提供する抽象データ型がある場合**<br>＋ 0.19 × unld_func 文数<br>＋（↑平均 unld_func 文長÷ 1024 ↑× unld_func 文数）<br>＋（↑平均 reld_func 文長÷ 1024 ↑× reld_func 文数） |
| DS | 1455※4＋ 32 ＋ 0.33 ×移動先 BES 数＋ 0.3 ×移動元 BES 数<br>＋ 0.2 ×移動先 RD エリア数＋ 0.22 ×移動元 RD エリア数＋ 0.34 × FES 数<br>＋（0.09 ＋↑平均 index 文ファイル長÷ 1024 ↑）× index 文数<br>＋（0.02 ＋↑平均ディレクトリ長÷ 1024 ↑）×（idxwork 文数＋ sort 文数）＋ 0.05 ×列数＋ 0.05 × RD エリア数＋ 0.15 ×インデクス数<br>＋ 0.05 ×インデクス格納 RD エリア数<br>**●対象表に LOB 列がある場合**<br>＋ 0.08 × LOB 格納 RD エリア数<br>**●対象表に BINARY 列がある場合**<br>＋ 33 ×（BINARY 列数×移動先 RD エリア数×移動元 RD エリア数）<br>**●対象表にプラグインが提供する抽象データ型がある場合**<br>＋ 0.01 ×移動先 BES 数＋ 0.19 × unld_func 文数<br>＋（↑平均 unld_func 文長÷ 1024 ↑× unld_func 文数）<br>＋（↑平均 reld_func 文長÷ 1024 ↑× reld_func 文数）<br>＋抽象データ型列数× 1 ＋（LOB 属性数× 0.05）× RD エリア数<br>＋データ型プラグイン数× 10 ＋プラグインインデクス数× 10 |
| BES | 6601※5＋ 50 ＋（517 ＋ 0.01 ×列数）×移動先 BES 数<br>＋（33 ＋ 0.01 ×列数）×移動元 BES 数＋ 0.2 ×移動先 RD エリア数<br>＋ 1.7 ×移動元 RD エリア数＋ 0.01 ×列数<br>＋ 0.26 ×移動先 RD エリア数×インデクス数<br>＋（0.09 ＋↑平均 index 文ファイル長÷ 1024 ↑）× index 文数<br>＋（0.02 ＋↑平均ディレクトリ長÷ 1024 ↑）×（idxwork 文数＋ sort 文数）<br>＋ 0.05 ×列数＋ 0.05 × RD エリア数＋ 0.15 ×インデクス数<br>＋ 0.05 ×インデクス格納 RD エリア数<br>＋ 550 × 1024 ＋ソートワークサイズ※1<br>**● -n オプションを指定した場合**<br>＋ RD エリアのページ長※2 ×一括入出力バッファ面数× y<br>**●対象表に LOB 列がある場合**<br>＋ 32 ＋ 0.01 × LOB 列数＋（32 ＋ 0.01 × LOB 列数）×移動先 BES 数<br>＋ 0.18 ×移動先 RD エリア数＋ 0.1 ×移動元 RD エリア数<br>＋ 0.08 × LOB 格納 RD エリア数<br>**●対象表に BINARY 列がある場合**<br>＋ 33 ×（BINARY 列数×移動先 RD エリア数×移動元 RD エリア数） |

| 条件 | メモリ所要量の計算式（単位：キロバイト） |
|---|---|
| | **●対象表にプラグインが提供する抽象データ型がある場合**<br>＋ 40 ＋（0.27 ＋ 2 ×抽象データ型長）×抽象データ型の列数<br>＋ 0.3 × unld_func 指定数<br>＋｛(64 ＋ 0.05 × LOB 属性数）×抽象データ型列数｝×移動先 BES 数<br>＋｛(64 ＋ 0.01 × unld_func 指定関数数＋ 0.07 ×抽象データ型属性数<br>＋ 0.05 × LOB 属性数）×抽象データ型列数｝×移動元 BES 数<br>＋（33 × BINARY 属性数×移動先 RD エリア数×移動元 RD エリア数）× 2<br>＋ 0.01 ×プラグインインデクス数＋ 0.19 × unld_func 文数<br>＋（↑平均 unld_func 文長÷ 1024 ↑× unld_func 文数）<br>＋（↑平均 reld_func 文長÷ 1024 ↑× reld_func 文数）<br>＋抽象データ型列数× 1 ＋（LOB 属性数× 0.05）× RD エリア数<br>＋データ型プラグイン数× 10 ＋プラグインインデクス数× 10<br>＋プラグインのメモリ所要量<br><br>**●インデクス作成方法に，インデクス一括作成モード又はインデクス情報出力モードを指定していて，次の条件を満たす場合**<br>**・表分割数×インデクス定義数＞プロセスのオープン数の上限** -576<br>＋ 2048 |

y：次に示すどちらかの値を代入してください。

- FIX ハッシュ分割表にリバランス機能を使用した場合
  （↑ 1024 ÷表全体の格納 RD エリア数↑）×該当サーバ内の表格納 RD エリア数
- そのほかの場合
  1

注※ 1
インデクスの一括作成時（-i c 又は省略）に加算します。

注※ 2
表を横分割した RD エリアごとにページ長が異なる場合は，ページ長の最大値で計算してください。

注※ 3
64 ビットモードの場合は 4660 です。

注※ 4
64 ビットモードの場合は 15830 です。

注※ 5
64 ビットモードの場合は 11804 です。

## 19.2.14 空きページ解放ユティリティ（pdreclaim）及びグローバルバッファ常駐化ユティリティ（pdpgbfon）実行時のメモリ所要量

空きページ解放ユティリティ（pdreclaim）及びグローバルバッファ常駐化ユティリティ（pdpgbfon）実行時のメモリ所要量は，次に示す計算式で求めます。

| 条件 | メモリ所要量の計算式（単位：キロバイト） |
|---|---|
| HiRDB/ シングルサーバの場合（32 ビットモードの場合） | 800 ＋ W |
| HiRDB/ シングルサーバの場合（64 ビットモードの場合） | 1897 ＋ W |

| 条件 | | メモリ所要量の計算式（単位：キロバイト） |
|---|---|---|
| HiRDB/ パラレルサーバの場合（32 ビットモードの場合） | システムマネジャ | 800 ＋ X |
| | -s オプションで指定したサーバ※1 | Y |
| | バックエンドサーバ※2 | Z |
| HiRDB/ パラレルサーバの場合（64 ビットモードの場合） | システムマネジャ | 1953 ＋ X |
| | -s オプションで指定したサーバ※1 | Y |
| | バックエンドサーバ※2 | Z |

注※ 1

-s オプション省略時は，処理対象表格納 RD エリアを 1 個目に定義したサーバです。

注※ 2

バックエンドサーバが複数ある場合は，バックエンドサーバ数だけメモリ所要量を加算します。

W：データベース再編成ユティリティ（pdrorg）実行時のシングルサーバのメモリ所要量

X：データベース再編成ユティリティ（pdrorg）実行時の MGR のメモリ所要量

Y：データベース再編成ユティリティ（pdrorg）実行時の DS のメモリ所要量

Z：データベース再編成ユティリティ（pdrorg）実行時の BES のメモリ所要量

データベース再編成ユティリティ（pdrorg）実行時のメモリ所要量については，「19.2.4　データベース再編成ユティリティ（pdrorg）実行時のメモリ所要量」を参照してください。

## 19.2.15　整合性チェックユティリティ（pdconstck）実行時のメモリ所要量

整合性チェックユティリティ（pdconstck）実行時のメモリ所要量は，次に示す計算式で求めます。

| 条件 | | | メモリ所要量の計算式（単位：キロバイト） |
|---|---|---|---|
| HiRDB/ シングルサーバの場合 | 32 ビットモードの場合 | | 11664 ＋ α |
| | 64 ビットモードの場合 | | 22696 ＋ α |
| HiRDB/ パラレルサーバの場合 | 32 ビットモードの場合 | MGR | 7577 ＋ α |
| | | DS | 5772 ＋ β |
| | 64 ビットモードの場合 | MGR | 9240 ＋ α |
| | | DS | 12776 ＋ β |

α：次に示す計算式から求められる値

2175

＋ 0.14 ×列数

＋ 0.09 ×表格納 RD エリア数

＋ 0.23 ×インデクス数

＋ 0.09 ×インデクス格納 RD エリア数

＋ 0.09 × LOB 列格納 RD エリア数

外部キー数
＋ Σ（42 ＋ 0.47 ×外部キー構成列数 r）
r=1
検査制約数
＋ Σ（5 ＋ 0.29 ×探索条件中の列数 c ＋ 0.85
c=1
×探索条件中の AND 及び OR の数 c ＋探索条件長 c）

β：次に示す計算式から求められる値
0.2
＋ 0.02 ×表格納 RD エリア数
＋ 0.02 ×インデクス格納 RD エリア数
＋ 0.02 × LOB 列格納 RD エリア数

# 20 リソース数に関連する環境変数の見積もり

この章では，リソース数に関連する環境変数の見積もり方法について説明します。

20.1 HiRDB/シングルサーバの場合
20.2 HiRDB/パラレルサーバの場合

# 20.1 HiRDB/シングルサーバの場合

リソース数に関連する環境変数の設定について説明します。これらの環境変数を設定することで，Windows にはないメッセージキュー，セマフォ，及び共用メモリの機能が使えるようになります。

環境変数の設定値の見積もりはユニット単位で行います。ただし，設定方法には OS 単位とユニット単位の二つがあります。HiRDB の構成に応じて設定方法を選択してください。

- **OS 単位で設定する**
  Windows のシステム環境変数として設定します。この場合，OS 内のすべてのユニットに対して同じ値が設定されます。OS 内に複数のユニットが存在するマルチ HiRDB 構成の場合は，各ユニットの見積もりの最大値を設定します。そのため，最大値を必要としないユニットはむだなリソースを使用することになります。
  OS 内に一つのユニットしか存在しない場合は OS 単位の設定で問題ありません。
  設定を有効にするためには，OS を再起動する必要があります。
- **ユニット単位で設定する**
  pdntenv コマンドの -sr オプションで設定します。OS 内に複数のユニットが存在する場合，各ユニットに適した値を設定できます。
  設定を有効にするためには，サービスを再開始する必要があります。

なお，両方で設定した場合，ユニット単位の設定値が優先されます。

## (1) 見積もり式

見積もり式を次の表に示します。

**表 20-1 リソース数に関連する環境変数に設定する値（HiRDB/シングルサーバの場合）**

| 種別 | 計算式 | 対応する環境変数 | 設定範囲 | 省略値 |
|---|---|---|---|---|
| メッセージキュー識別子数（単位：個） | （16 ＋ f）× a ＋ 28 ＋ 1 | OS 単位：システム環境変数 PDUXPLMSGMNI<br>ユニット単位：`pdntenv -sr msgmni` | 50 ～ 3,600,000 ※1, 2 | ● 32 ビットモードの場合<br>50<br>● 64 ビットモードの場合<br>100 |
| メッセージキューテーブル数（単位：個） | b ×ユニット数＋ A ＋ B | OS 単位：システム環境変数 PDUXPLMSGTQL ユニット単位：`pdntenv -sr msgtql` | 80 ～ 2,048 ※1, 2 | ● 32 ビットモードの場合<br>80<br>● 64 ビットモードの場合<br>2,048 |
| セマフォ識別子数（単位：個） | {↑ {2 ×（b ＋ 3）＋ 12} ÷ 64 ↑＋↑ c ÷ 64 ↑＋ g ＋ 5} × a ＋ 2 ＋ d | OS 単位：システム環境変数 PDUXPLSEMMAX ユニット単位：`pdntenv -sr semmax` | 64 ～ 2,147,483,647 ※1 | 64 |
| 共用メモリ使用数（単位：セグメント） | （2 ＋ h ＋ pd_max_add_dbbuff_shm_no の値※3 ＋ pd_max_resident_rdarea_shm_no の値※4）×（e ＋ i ＋ 50） | OS 単位：システム環境変数 PDUXPLSHMMAX ユニット単位：`pdntenv -sr shmmax` | 4,096 ～ 2,147,483,647 ※1 | ● 32 ビットモードの場合<br>4,096<br>● 64 ビットモードの場合<br>16,384 |

注※1
設定範囲より小さい値を設定した場合，HiRDB が最小値に切り上げます。

注※ 2

設定範囲より大きい値を設定した場合，HiRDB が最大値に切り下げます。

注※ 3

グローバルバッファの動的変更機能を使用する場合に加算してください。セキュリティ監査機能使用時はさらに 1 を加算してください。

注※ 4

インメモリデータ処理を行う場合に加算してください。

A：次の値を代入してください。

```
N
Σ {
i=1
    サーバiに割り当てたグローバルバッファプール数※1
    ＋サーバiのシンクポイントダンプ有効化のスキップ回数※2×2
    ＋サーバiで実行するpdload，pdrorg，pdrbal，ログレスUAPの最大同時実行数＋1
}

N：1
```

注※ 1

pdbufls コマンドで表示されるバッファプール名をサーバ単位に集計することで確認できます。

注※ 2

pd_spd_syncpoint_skip_limit に 0 以外の値を指定している場合は pd_spd_syncpoint_skip_ limit 指定値で見積もります。

pd_spd_syncpoint_skip_limit を省略しているか，0 を指定している場合は，マニュアル「HiRDB Version 8 システム運用ガイド」「8.8　UAP の状態監視　(3) 計算方法　(b) 全システムログファイルの容量から計算する方法」を参照し見積もってください。

B：非同期 READ 機能使用時（pd_max_ard_process に 0 以外指定時）だけ加算します。非同期 READ 機能を使用しない場合は 0 で見積もります。次の値を代入してください。

```
N
Σ {
i=1
    サーバiに割り当てたグローバルバッファプールのpdbuffer -m指定値の合計
}

N：1
```

**a**：シングルサーバの場合は 1

**b**：pd_max_users オペランドの値

**c**：pdbuffer オペランドの定義数

**d**：系切り替え機能使用時に加算します。次に示す表から値を求めてください。

| 条件 | | | d の値 |
|---|---|---|---|
| pd_ha_acttype=monitor（又は省略） | | | 0 |
| pd_ha_acttype=server | pd_ha_agent=standbyunit | | 1 |
| | pd_ha_agent を省略 | pd_ha_server_process_standby=Y（又は省略） | 1 |
| | | pd_ha_server_process_standby=N | 0 |

**e**：pd_max_server_process オペランドの値

**f**：1（pd_max_ard_process オペランドに 1 以上を指定する場合）又は 0

**g**：2（pd_dfw_awt_process オペランドに値を指定する場合）又は 0

h：↑（グローバルバッファが使用する共用メモリの総量（※）÷ SHMMAX の値）↑

i：pd_aud_file_name オペランドを指定している場合は b の値，pd_aud_file_name オペランドを指定していない場合は 0

注※

「15.1.5　グローバルバッファが使用する共用メモリの計算式」を参照してください。

## （2）共用メモリの計算式

メッセージキュー，セマフォ，及び共用メモリの機能を使うことによって使用される共用メモリの計算式を次の表に示します。

**表 20-2　共用メモリの計算式**

| 共用メモリを使う機能 | 計算式（単位：バイト） |
|---|---|
| メッセージキュー | **● 32 ビットモードの場合**<br>((((16384 ＋（メッセージキューテーブル数× 24)) ＋ 15)<br>÷ 16）× 16 ＋ 384）<br>＋（(メッセージキュー識別子数− 1）× 64）<br>＋（(メッセージキューテーブル数− 1）× 32）<br>**● 64 ビットモードの場合**<br>((((16384 ＋（PDUXPLMSGTQL × 24)) ＋ 15)<br>÷ 16）× 16 ＋ 408）<br>＋（(メッセージキュー識別子数− 1）× 80）<br>＋（(メッセージキューテーブル数− 1）× 40） |
| セマフォ | **● 32 ビットモードの場合**<br>64 ＋セマフォ識別子数の設定値× 176<br>**● 64 ビットモードの場合**<br>104 ＋セマフォ識別子数の設定値× 184 |
| 共用メモリ（共用メモリを使用する場合，管理用の共用メモリが必要になる） | **● 32 ビットモードの場合**<br>16 ＋共用メモリ使用数の設定値× 40<br>**● 64 ビットモードの場合**<br>24 ＋共用メモリ使用数の設定値× 64 |

# 20.2　HiRDB/ パラレルサーバの場合

リソース数に関連する環境変数の設定について説明します。これらの環境変数を設定することで，Windows にはないメッセージキュー，セマフォ，及び共用メモリの機能が使えるようになります。

環境変数の設定値の見積もりはユニット単位で行います。ただし，設定方法には OS 単位とユニット単位の二つがあります。HiRDB の構成に応じて設定方法を選択してください。

**● OS 単位で設定する**

Windows のシステム環境変数として設定します。この場合，OS 内のすべてのユニットに対して同じ値が設定されます。OS 内に複数のユニットが存在するマルチ HiRDB 構成の場合は，各ユニットの見積もりの最大値を設定します。そのため，最大値を必要としないユニットはむだなリソースを使用することになります。

OS 内に一つのユニットしか存在しない場合は OS 単位の設定で問題ありません。

設定を有効にするためには，OS を再起動する必要があります。

**● ユニット単位で設定する**

pdntenv コマンドの -sr オプションで設定します。OS 内に複数のユニットが存在する場合，各ユニットに適した値を設定できます。

設定を有効にするためには，OS を再起動する必要があります。

なお，両方で設定した場合，ユニット単位の設定値が優先されます。

## (1) 見積もり式

見積もり式を次の表に示します。

**表 20-3　リソース数に関連する環境変数に設定する値（HiRDB/ パラレルサーバの場合）**

| 種別 | 計算式 | 対応する環境変数 | 設定範囲 | 省略値 |
|---|---|---|---|---|
| メッセージキュー識別子数（単位：個） | $b \sum_{i=1} V_i + 2 \times a + 3 \times b + c + d + e + 25 + 1 + m$ | OS 単位：システム環境変数 PDUXPLMSGMNI<br>ユニット単位：`pdntenv -sr msgmni` | 50 ～ 3,600,000 ※1, 2 | ● 32 ビットモードの場合<br>50<br>● 64 ビットモードの場合<br>100 |
| メッセージキューテーブル数（単位：個） | h ×ユニット数＋ A ＋ B | OS 単位：システム環境変数 PDUXPLMSGTQL<br>ユニット単位：`pdntenv -sr msgtql` | 80 ～ 2,048 ※1, 2 | ● 32 ビットモードの場合<br>80<br>● 64 ビットモードの場合<br>2,048 |

| 種別 | 計算式 | 対応する環境変数 | 設定範囲 | 省略値 |
|---|---|---|---|---|
| セマフォ識別子数<br>（単位：個） | **●影響分散スタンバイレス型系切り替え機能を使用していない場合**<br>$\sum_{i=1}^{b} \{\uparrow (S_i + T_i + U_i) \div 64 \uparrow + W_i\}$<br>$+ 6 \times b + 2 + f$<br><br>**●影響分散スタンバイレス型系切り替え機能を使用している場合**<br>$\sum_{i=1}^{b} \{\uparrow \{Y_i \times (j + k)\} \div 64 \uparrow + W_i\}$<br>$+ 6 \times b + 2 + f$ | OS 単位：システム環境変数<br>PDUXPLSEMMAX ユニット単位：<br>`pdntenv -sr semmax` | 64 ～<br>2,147,483,647<br>※ 1 | 64 |
| 共用メモリ使用数<br>（単位：セグメント） | $(2 + z + \sum_{i=1}^{n}$ 各サーバに指定した pd_max_add_dbbuff_shm_no の値※ 3<br>$+$ (pd_max_resident_rdarea_shm_no の値×バックエンドサーバ数) ※ 4<br>$) \times (g + Ai + 50)$ | OS 単位：システム環境変数<br>PDUXPLSHMMAX ユニット単位：<br>`pdntenv -sr shmmax` | 4,096 ～<br>2,147,483,647<br>※ 1 | ● 32 ビットモードの場合<br>4,096<br>● 64 ビットモードの場合<br>16,384 |

注※ 1

設定範囲より小さい値を設定した場合，HiRDB が最小値に切り上げます。

注※ 2

設定範囲より大きい値を設定した場合，HiRDB が最大値に切り下げます。

注※ 3

グローバルバッファの動的変更機能を使用する場合に加算してください。セキュリティ監査機能使用時はさらに 1 を加算してください。

注※ 4

インメモリデータ処理を行う場合に加算してください。

A：次の値を代入してください。

```
N
Σ {
i=1
    サーバiに割り当てたグローバルバッファプール数※1
    ＋サーバiのシンクポイントダンプ有効化のスキップ回数※2×2
    ＋サーバiで実行するpdload，pdrorg，pdrbal，ログレスUAPの最大同時実行数＋1
}

N：
    ユニット内のバックエンドサーバ数＋ユニット内のディクショナリサーバ数＋ユニット内のゲスト BES 数※3

注※1
    pdbufls コマンドで表示されるバッファプール名をサーバ単位に集計することで確認できます。
注※2
    pd_spd_syncpoint_skip_limit に 0 以外の値を指定している場合は pd_spd_syncpoint_skip_ limit 指定値で見積もります。
    pd_spd_syncpoint_skip_limit を省略しているか，0 を指定している場合は，マニュアル「HiRDB Version 8 システム運用ガイド」「8.8　UAP の状態監視　(3) 計算方法　(b) 全システムログファイルの容量から計算する方法」を参照し見積もってください。
注※3
影響分散スタンバイレス型系切り替え適用ユニットの場合に加算します。
```

**B**：非同期 READ 機能使用時（pd_max_ard_process に 0 以外指定時）だけ加算します。非同期 READ 機能を使用しない場合は 0 で見積もります。次の値を代入してください。

```
N
Σ {
i=1
    サーバiに割り当てたグローバルバッファプールのpdbuffer -m指定値の合計
}

N：
    ユニット内のバックエンドサーバ数＋ユニット内のディクショナリサーバ数＋ユニット内のゲスト BES 数※

注※
影響分散スタンバイレス型系切り替え適用ユニットの場合に加算します。
```

**a**：サーバマシン内のフロントエンドサーバ数

**b**：サーバマシン内のディクショナリサーバ及びバックエンドサーバ数

**c**：フロントエンドサーバの場合は 4，それ以外は 0

**d**：ディクショナリサーバの場合は 8，それ以外は 0

**e**：バックエンドサーバの場合は 16，それ以外は 0

**f**：系切り替え機能使用時に加算します。次に示す表から値を求めてください。

| 条件 | | | f の値 |
|---|---|---|---|
| pd_ha_acttype=monitor（又は省略） | | | 0 |
| pd_ha_acttype=server | pd_ha_agent=standbyunit | | 1 |
| | pd_ha_agent を省略 | pd_ha_server_process_standby=Y（又は省略） | 1 |
| | | pd_ha_server_process_standby=N | 0 |

**g**：pd_max_server_process オペランドの値

**h**：pd_max_users オペランドの値

**j**：ホスト BES 数

**k**：ゲスト BES 数

**m**：システムマネジャユニットがある場合は 3，ない場合は 0

z：↑（ユニット内のグローバルバッファが使用する共用メモリの総量（※）÷ SHMMAX の値）↑

A：pd_aud_file_name オペランドを指定している場合は h の値，pd_aud_file_name オペランドを指定していない場合は 0

Si：各サーバに配置する RD エリアに対する pdbuffer -r オペランドの定義数

Ti：各サーバに配置する RD エリアに対する pdbuffer -i オペランドの定義数

Ui：pdbuffer -o オペランドの定義数

Vi：1（pd_max_ard_process オペランドに 1 以上を指定する場合）又は 0

Wi：2（pd_dfw_awt_process オペランドに値を指定する場合）又は 0

$Y_i$：pdbuffer オペランドの -c オプションの指定数

注※

「15.2.5　グローバルバッファが使用する共用メモリの計算式」を参照してください。

## (2) 共用メモリの計算式

メッセージキュー，セマフォ，及び共用メモリの機能を使うことによって使用される共用メモリの計算式を次の表に示します。

**表 20-4　共用メモリの計算式**

| 共用メモリを使う機能 | 計算式（単位：バイト） |
|---|---|
| メッセージキュー | **● 32 ビットモードの場合**<br>((((16384 ＋（メッセージキューテーブル数× 24)) ＋ 15)<br>÷ 16) × 16 ＋ 384)<br>＋（(メッセージキュー識別子数− 1) × 64)<br>＋（(メッセージキューテーブル数− 1) × 32)<br>**● 64 ビットモードの場合**<br>((((16384 ＋（メッセージキューテーブル数× 24)) ＋ 15)<br>÷ 16) × 16 ＋ 408)<br>＋（(メッセージキュー識別子数− 1) × 80)<br>＋（(メッセージキューテーブル数− 1) × 40) |
| セマフォ | **● 32 ビットモードの場合**<br>64 ＋セマフォ識別子数の設定値× 176<br>**● 64 ビットモードの場合**<br>104 ＋セマフォ識別子数の設定値× 184 |
| 共用メモリ（共用メモリを使用する場合，管理用の共用メモリが必要になる） | **● 32 ビットモードの場合**<br>16 ＋共用メモリ使用数の設定値× 40<br>**● 64 ビットモードの場合**<br>24 ＋共用メモリ使用数の設定値× 64 |

# 21 Windows レジストリ設定値の見積もり

この章では，Windows レジストリ設定値の見積もり方法について説明します。

## 21.1 デスクトップヒープ指定値の見積もり

# 21.1 デスクトップヒープ指定値の見積もり

HiRDB は，必要なデスクトップヒープ量を計算し，ユニット起動時に自動的に確保します。そのため，通常はデスクトップヒープの指定値を変更する必要はありません。ただし，デスクトップヒープ不足が頻発する場合は，次の方法でデスクトップヒープの指定値を見直してください。

## (1) デスクトップヒープの指定値の求め方

デスクトップヒープの使用量はレジストリ中に指定があり，レジストリエディタで変更できます。この指定はアカウントごとの使用量を規定するもので，サーバマシン全体で 48 メガバイトが上限です。

**計算式**

HiRDB が起動するサーバプロセス数（pd_max_server_process の値）× a　（単位：バイト）

a の値：

Windows 2000，Windows XP，又は Windows Vista の場合は 5000

Windows Server 2003 の場合は 100

ただし，次の点に注意が必要です。

- 指定値を大きくし過ぎると，別のアカウントで稼働している他 PP の動作に影響を与える可能性があります。このため，不要に大きな値を指定しないでください。
- HiRDB が起動するプロセスは，HiRDB サーバプロセスやユーザが実行する運用コマンド，及びユティリティだけではありません。HiRDB のサーバプロセスが異常終了した後に実行する保守情報を取得するコマンドも，HiRDB が起動するプロセスとしてデスクトップヒープを使用します。このため，大量の HiRDB サーバプロセス異常終了又はキャンセル時に，デスクトップヒープ不足（サーバプロセス停止を示す KFPS01820-E メッセージで「end state=0x8000」と表示されます）が発生した場合，デスクトップヒープ量を調整するのではなく，システム共通定義で保守情報を取得しないように変更してください。
- システムアカウントで HiRDB を稼働させている場合，OS などほかのシステムアカウントで動作しているプログラムと同じデスクトップヒープを使用します。このため，例えば Windows 2000 を使用し，デスクトップヒープの使用量が 512 キロバイトの場合，HiRDB が起動しているプロセスが 100 未満でも，HiRDB のプロセス起動時にデスクトップヒープ不足を検出する場合があります。

## (2) デスクトップヒープの指定値の変更

### (a) HiRDB のシステム定義で変更する場合

システム共通定義，又はユニット制御情報定義の pd_process_desktopheap_size オペランドで，1 プロセス当たりのデスクトップヒープ消費量を指定します。

### (b) レジストリエディタで変更する場合

上記の方法でデスクトップヒープ不足が解消されない場合は，レジストリエディタでデスクトップヒープの指定値を変更します。なお，レジストリエディタの使い方を誤ると重大な問題が発生することがあります。十分注意してください。また，デスクトップヒープの値は，動作環境に応じて調整する必要があります。

1. HiRDB のサービスを停止します。
2. レジストリエディタを使用して，非対話型デスクトップのデスクトップヒープの値を変更します。

**●レジストリキー**

HKEY_LOCAL_MACHINE¥SYSTEM¥CurrentControlSet¥Control¥Session Manager¥SubSystems

**●レジストリ値**

Windows

**●値**

%SystemRoot%¥system32¥csrss.exe ObjectDirectory =¥Windows SharedSection=1024,3072,512 Windows=On SubSystemType=Windows ServerDll=basesrv,1 ServerDll=winsrv:UserServerDllInitialization,3 ServerDll=winsrv:ConServerDllInitialization,2 ProfileControl=Off MaxRequestThreads=16

変更するパラメタは，SharedSection の 3 番目のパラメタ（下線）です。このパラメタが 512 の場合は，システムは各デスクトップに対して 512 キロバイトのヒープを割り当てます。3 番目のパラメタが省略されている場合は，2 番目の値（3,072 キロバイト）を割り当てます。3 番目のパラメタを「(1) デスクトップヒープの指定値の求め方」で求めた値に変更してください。この場合，pd_max_server_process の指定値が正しく見積られた値であることが前提です。

3. OS を再起動します。
4. HiRDB サービスを再開始します。

# 22 サンプルファイル

この章では，HiRDB が提供するサンプルファイル（サンプルデータベース，サンプルコンフィグレーション及びサンプル UOC）について説明します。


22.1 **サンプルファイルの概要**

22.2 **表の定義情報**

22.3 **サンプルファイルの使用方法**

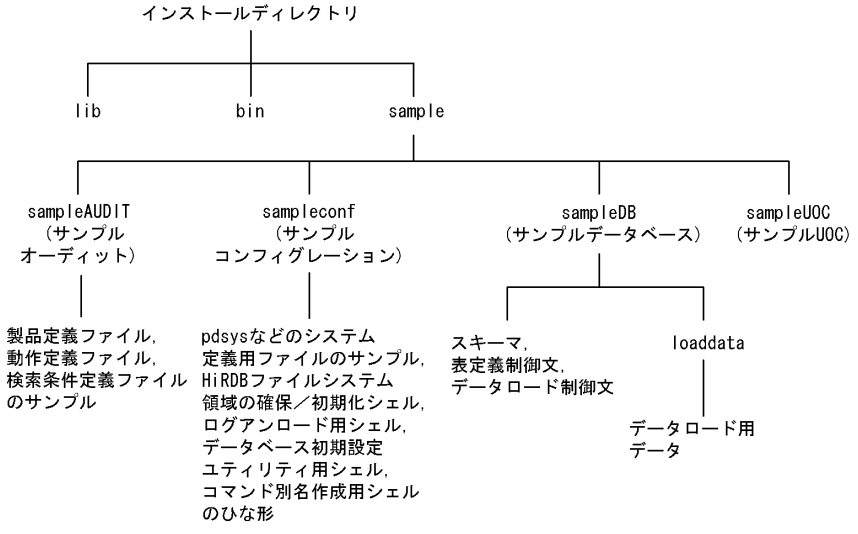
# 22.1　サンプルファイルの概要

HiRDB が提供するサンプルファイルを次に示します。

- サンプルオーディット
- サンプルデータベース
- サンプルコンフィグレーション
- サンプル UOC

サンプルファイルのディレクトリ構成を次の図に示します。なお，sample 以下のディレクトリは，インストールディレクトリの下にあります。

**図 22-1　サンプルファイルのディレクトリ構成**




以降の説明は，バッチファイル SPsetup.bat を実行して HiRDB の環境を設定し終わったことを前提に説明します。SPsetup.bat については，「付録 D　バッチファイルによる環境設定」を参照してください。

## 22.1.1　サンプルファイルのファイル名

ここでは，下記のサンプルのファイル名について説明します。

- サンプルオーディット
- サンプルデータベース
- サンプルコンフィグレーション
- サンプル UOC

### (1) サンプルオーディットのファイル名

サンプルオーディットは，JP1/NETM/Audit との連携で使用するサンプルファイルです。ファイル名とその内容を次の表に示します。

**表 22-1 サンプルオーディットのファイル名と内容**

| ファイル名 | 内容 |
|---|---|
| HiRDB.conf | 製品定義ファイル |
| admjevlog_HiRDB.conf | 動作定義ファイル |
| sampleaud1 | 検索条件定義ファイル |
| sampleaud2 | |

これらのサンプルファイルを使用した環境設定については，マニュアル「HiRDB Version 8 システム運用ガイド」の「JP1/NETM/Audit との連携」を参照してください。

## (2) サンプルデータベースのファイル名

サンプルで使用するディレクトリ及びファイル名とその内容を次の表に示します。

**表 22-2 サンプルで使用するディレクトリ及びファイル名とその内容**

| ディレクトリ名<br>又はファイル名 | 内容 | 備考 |
|---|---|---|
| tblecreate | 表定義文（スキーマ定義を含む） | pddef 入力形式 |
| loaddata | データロード入力データ | ディレクトリ |
| SEIGYO_FILE | データロード用制御文 | ― |

## (3) サンプルコンフィグレーションのファイル名

サンプルコンフィグレーションの内容を次の表に示します。

ここで示す内容は，パラメタ間の関連性などを分かりやすくするために最小構成での指定値を例として示したもので，最適な値を示したものではありません。

**表 22-3 サンプルコンフィグレーションの内容**

| 分類 | 内容 | ファイル名 ※1 |
|---|---|---|
| システム定義 | システム共通定義 | pdsys |
| | ユニット制御情報定義 | pdutsys |
| | シングルサーバ定義 | sds01 |
| HiRDB ファイルシステム領域の確保／初期化 | 次に示す HiRDB ファイルシステム領域を作成するバッチファイル<br>• RD エリア用の HiRDB ファイルシステム領域<br>• 作業表用ファイル用の HiRDB ファイルシステム領域 | fmkfile.bat |
| | 次に示す HiRDB ファイルシステム領域を作成するバッチファイル<br>• システムファイル用の HiRDB ファイルシステム領域 | fmkfs.bat |
| | システムログ，シンクポイントダンプ及びステータスファイル初期化バッチファイル | sysfint.bat |
| ログアンロード | システムログファイルアンロード用バッチファイル | logunld.bat ※2 |
| データベース初期設定ユティリティ<br>（pdinit） | RD エリアを作成するバッチファイルです。データベース初期設定ユティリティ（pdinit）を実行するバッチファイル | initdb.bat |
| | データベース初期設定ユティリティ（pdinit）の制御文を格納しているバッチファイル | mkinit |
| コマンドの別名での実行 | コマンドを別名で実行するための実行形式ファイルのひな形（バッチファイル） | aliascmd.bat ※2 |

注※ 1

ここでは HiRDB/ シングルサーバの場合のファイル名を示します。HiRDB/ パラレルサーバの場合は，%PDDIR%¥HiRDEF¥readme.txt を参照してください。

注※ 2

logunld.bat は %PDDIR%¥sample ディレクトリ下に，aliascmd.bat は %PDDIR%¥sample¥sampleconf ディレクトリ下に格納されます。

## (4) サンプル UOC のファイル名

次に示す UOC を格納しています。サンプル UOC の内容を次の表に示します。

- データベース作成ユティリティ（pdload）のファイル入力の例
- データベース再編成ユティリティ（pdrorg）のファイル出力の例

UOC については，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

**表 22-4 サンプル UOC の内容**

| ファイル名 | 内容 |
|---|---|
| sample1.c | DAT 形式入力ファイルをデータベース作成ユティリティ（pdload）が入力する UOC の例です。 |
| sampleA.c | 不要なデータをアンロードファイルに出力しない UOC の例です。 |

# 22.2 表の定義情報

サンプルデータベースとして提供している表の種類を表 22-5 に示します。また，その表の列属性を表 22-6 に示します。

これらの表はすべて FIX **属性の表**です。さらに，バッチファイル SPsetup.bat をカスタマイズしないで実行すると，これらの表は**ユーザ用** RD **エリアの** RDDATA10 に格納されます。インデクスは**ユーザ用** RD **エリアの** RDINDX10 に格納されます。

**表** 22-5 **サンプルデータベースとして提供されている表**

| 表名 | 内容 | 行数 |
|---|---|---|
| CUSTOM | 得意先マスタ | 100 |
| GOODS | 商品マスタ | 100 |
| VENDOR | 仕入先マスタ | 50 |
| TAKEODR | 受注 | データロード対象外 |
| STOCK | 在庫 | 100 |
| WAREHUS | 入庫 | データロード対象外 |
| SHIPMNT | 出庫 | データロード対象外 |
| SENDODR | 発注 | データロード対象外 |
| LAYIN | 仕入れ | データロード対象外 |

**表** 22-6 **表の列属性及びインデクス**

| 表名 | 列名 | 列属性 | インデクス名 |
|---|---|---|---|
| CUSTOM | トクイサキ CD | CHAR(5) | UNIQUE CLUSTER KEY CUSTOMX |
| | トクイサキメイ | CHAR(30) | |
| | TELNO | CHAR(12) | |
| | ZIPCD | CHAR(3) | |
| | ジユウシヨ | CHAR(30) | |
| GOODS | シヨウヒン CD | CHAR(6) | UNIQUE CLUSTER KEY GOODSX |
| | シヨウヒンメイ | CHAR(30) | |
| | タンカ | DECIMAL(7,0) | |
| | シイレサキ CD | CHAR(5) | |
| VENDOR | シイレサキ CD | CHAR(5) | UNIQUE CLUSTER KEY VENDORX |
| | シイレサキメイ | CHAR(30) | |
| | TELNO | CHAR(12) | |
| | ZIPCD | CHAR(3) | |
| | ジユウシヨ | CHAR(30) | |

| 表名 | 列名 | 列属性 | インデクス名 |
|---|---|---|---|
| TAKEODR | ジユチユウ CD | CHAR(7) | CLUSTER KEY |
| | トクイサキ CD | CHAR(5) | |
| | シヨウヒン CD | CHAR(6) | |
| | スウリヨウ | DECIMAL(7,0) | |
| | ヒキアテリヨウ | DECIMAL(7,0) | |
| | ジユチユウザン | DECIMAL(7,0) | |
| | ジユチユウヒヅケ | CHAR(6) | |
| | ノウキ | CHAR(6) | |
| STOCK | シヨウヒン CD | CHAR(6) | CLUSTER KEY |
| | ザイコリヨウ | DECIMAL(7,0) | |
| | ヒキアテリヨウ | DECIMAL(7,0) | |
| | ハツチユウテン | DECIMAL(7,0) | |
| | シイレサキ CD | CHAR(5) | |
| WAREHUS | シヨウヒン CD | CHAR(6) | CLUSTER KEY |
| | ニユウコリヨウ | DECIMAL(7,0) | |
| | シイレ NO | INTEGER | |
| | ニユウコビ | CHAR(6) | |
| SHIPMNT | シヨウヒン CD | CHAR(6) | CLUSTER KEY |
| | シユツコリヨウ | DECIMAL(7,0) | |
| | ジユチユウ CD | CHAR(7) | |
| | ジユチユウビ | CHAR(6) | |
| SENDODR | ハツチユウ NO | INTEGER | CLUSTER KEY<br>SENDODRX |
| | シイレサキ CD | CHAR(5) | |
| | シヨウヒン CD | CHAR(6) | |
| | ハツチユウリヨウ | DECIMAL(7,0) | |
| | ハツチユウビ | CHAR(6) | |
| | ノウキ | CHAR(6) | |
| LAYIN | シイレ NO | INTEGER | CLUSTER KEY<br>LAYINX |
| | シイレサキ CD | CHAR(5) | |
| | シヨウヒン CD | CHAR(6) | |
| | シイレリヨウ | DECIMAL(7,0) | |
| | シイレビ | CHAR(6) | |

# 22.3 サンプルファイルの使用方法

サンプルデータベースを作成するバッチファイルを次の表に示します。これらのバッチファイルは%PDDIR%¥sample¥tools（例：C:¥win32app¥hitachi¥hirdb_s¥sample¥tools）ディレクトリ下にあります。

**表 22-7 サンプルデータベースを作成するバッチファイル**

| ファイル名 | 説明 |
|---|---|
| sampleDB1.bat<br>sampleDB2.bat | 表作成者を root にする場合に使用します。<br>• sampleDB1.bat はサンプルデータベースを定義するバッチファイルです。<br>• sampleDB2.bat はサンプルデータベースにデータをロードするバッチファイルです。 |
| sampleDB3.bat<br>sampleDB4.bat | 表作成者を USER1 にする場合に使用します。<br>• sampleDB3.bat は，USER1 に CONNECT 権限及びスキーマ定義権限を与えて，サンプルデータベースを定義するバッチファイルです。<br>• sampleDB4.bat はサンプルデータベースにデータをロードするバッチファイルです。 |

**注意事項**

- このバッチファイルの内容をカスタマイズする場合，ファイル名を変更しないで直接カスタマイズすると，HiRDB の更新インストール時に上書きされます。HiRDB が提供しているバッチファイルを別名でコピーしたものをカスタマイズして利用してください。
- 以降の説明は HiRDB/ シングルサーバをデフォルトのインストールディレクトリ（C:¥win32app¥hitachi¥hirdb_s）にインストールしたと仮定しています。必要に応じてパス名及び HiRDB の種類（HiRDB/ パラレルサーバ）を読み替えてください。
  また，HiRDB/ パラレルサーバの場合は，%PDDIR%¥HiRDEF¥readme.txt を参照してください。

## 22.3.1 サンプルデータベースの作成手順

サンプルデータベースの作成手順を次に示します。

1. **表の定義及びユーザを登録します**
   sampleDB1.bat をダブルクリックしてください。
   **「続行するときは何かキーを押してください」**と表示されるので，問題がなければ**【Enter キー】**を押してください。問題がある場合は問題を解決して，再実行してください。この場合，表作成者は「root」になります。表作成者を「USER1」にしたい場合は，sampleDB3.bat をダブルクリックしてください。

2. **表へデータをロードします**
   sampleDB2.bat をダブルクリックしてください。
   **「続行するときは何かキーを押してください」**と表示されるので，問題がなければ**【Enter キー】**を押してください。問題がある場合は問題を解決して，再実行してください。表作成者を「USER1」にした場合（sampleDB3.bat を使用した場合）は，sampleDB4.bat をダブルクリックしてください。

## 22.3.2 サンプルデータベースのカスタマイズ

サンプルデータベースを作成するバッチファイルをカスタマイズして，表の定義及び表へのデータロードに利用してください。カスタマイズするには，SQL の知識及びデータベース作成ユティリティ（pdload）の知識が必要になります。SQL についてはマニュアル「HiRDB Version 8 SQL リファレンス」を，データベース作成ユティリティ（pdload）についてはマニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

**注意事項**

インストールして HiRDB の開始が完了した時点で登録されている認可識別子とパスワードは「root」です。「root」ユーザは DBA 権限となっています。このため，サンプルデータベースの新規セットアップ実行時にクライアント環境定義（hirdb.ini）中の認可識別子とパスワードはそれぞれ「root」が設定されています（PDUSER="root" / "root"）。

## (1) 新しいユーザを追加する場合

新しいユーザを追加する場合のカスタマイズ方法を次に示します。

**カスタマイズ方法**

1. sampleDB1.bat の内容を変更します。表定義を実行している部分をコメント化して，権限定義だけが実行されるように変更します。

   ```
   pddef<%PDDIR%¥sample¥sampleDB¥tblecreate
                          ↓
   rem pddef<%PDDIR%¥sample¥sampleDB¥tblecreate
   ```

2. 権限定義のサンプルとして提供している %PDDIR%¥sample¥sampleDB¥gr_USER1 の内容を，次のように変更します。
   ・新しいユーザを追加するため，GRANT 文で CONNECT 権限を定義します。
   ・必要に応じて GRANT 文でスキーマ定義権限などを定義します。
3. 変更が終了したら，sampleDB1.bat をダブルクリックします。データベース定義ユティリティ（pddef）が正常に終了したことを示すメッセージが出力されると，新しいユーザの追加と権限定義は完了します。
   なお，sampleDB3.bat は「USER1」を追加する例です。

## (2) 表及びインデクスを定義する場合

表及びインデクスを定義する場合のカスタマイズ方法を次に示します。

**カスタマイズ方法**

1. 新しいユーザを追加してそのユーザでスキーマ定義などをする場合，クライアント環境定義（hirdb.ini）の認可識別子とパスワードを変更してください（環境変数 windir の直下にある hirdb.ini ファイル中の環境変数 PDUSER の設定値を「root」から，追加した認可識別子とパスワードに変更してください）。
2. sampleDB1.bat の内容を変更します。権限定義を実行している部分をコメント化し，表定義部分をコメント化していれば元に戻してください。
3. 表定義のサンプルとして提供している %PDDIR%¥sample¥sampleDB¥tblecreate の内容を変更します。定義するスキーマ又は表などの内容を変更します。
4. 変更が終了したら，sampleDB1.bat をダブルクリックします。データベース定義ユティリティ（pddef）が正常に終了したことを示すメッセージが出力されると完了です。複数の表を定義して，その中にエラーの表がある場合，tblecreate の内容を変更して，もう一度実行してください。

## (3) 表にデータロードをする場合

表にデータロードをする場合のカスタマイズ方法を次に示します。

**カスタマイズ方法**

1. sampleDB2.bat では認可識別子及びパスワードがそれぞれ「root」になっているので，対応する表の所有者の認可識別子及びパスワードに変更してください。

   ```
   set   PDUSER = "root"/"root"
           ↓
   ```

set PDUSER = "**認可識別子**"/"**パスワード**"

2. 提供しているサンプルでは四つの表（CUSTOM，GOODS，VENDOR，STOCK）にデータロードをするため，必要に応じてこれらを変更又はコメント化してください。
3. @echo source%PDDIR%¥sample¥sampleDB¥loaddata¥GOODS.CSV ＞ %PDDIR%¥TMP¥LOD で制御情報ファイル LOD に制御情報を設定しています。必要に応じて設定している制御情報部分を変更してください。
4. %PDDIR%¥bin¥pdload -i s -e GOODS %PDDIR%¥TMP¥LOD でデータベース作成ユティリティ（pdload）を実行しています。表名 GOODS など必要に応じて変更してください。
5. %PDDIR%¥sample¥sampleDB¥loaddata¥GOODS.csv が入力ファイルとなっています。ここで指定したファイル中に入力データを作成してください。提供しているサンプルデータでは，GOODS.CSV が DAT 形式になっているので，これを参考に入力データを作成してください。
6. 入力データの作成が終了したら，sampleDB2.bat をダブルクリックします。データベース作成ユティリティ（pdload）の実行状態を示すメッセージが出力されるので，正しく実行されているかどうかを確認してください。
   なお，sampleDB4.bat は「USER1」がデータロードを実行する場合の例です。

## （4） コマンドを別名で実行するためのバッチファイルの作成

HiRDB のコマンド名と OS やほかのプログラムが提供しているコマンド名が同一になり，HiRDB のコマンドが実行できない場合があります。このような場合，次に示す回避策があります。

- **環境変数の設定で** HiRDB のコマンドを優先します。
- **絶対パスを指定してコマンドを実行します。**

上記二つの回避策が実行できない場合に，HiRDB のコマンドを任意の名称で実行する方法があります。HiRDB ではこの方法を実現するために必要なバッチファイルのひな形を提供しています。

### （a） HiRDB が提供しているバッチファイルのひな形ファイル名

HiRDB が提供しているバッチファイルのひな形ファイルと内容を次の表に示します。ファイルの格納場所を次に示します。

- C:¥win32app¥hitachi¥hirdb_s¥sample¥sampleconf

**表 22-8 コマンド名の別名実行用バッチファイルのひな形ファイルと内容**

| ファイル名 | 内容 | 注意 |
|---|---|---|
| aliascmd.bat | バッチファイルのひな形ファイル | HiRDB 運用ディレクトリ下の bin 及び lib の二つのディレクトリ以下にはコピーしないでください。 |

### （b） コマンド名の別名を作成する手順

次に示す手順で，コマンドの別名を作成します。

1. バッチファイルのひな形ファイルを任意のディレクトリにコピーします。複数のコマンドの別名を作成したい場合は，コマンドの数だけコピーしてください。ただし，HiRDB 運用ディレクトリ下の bin 及び lib のディレクトリ以下にコピーしないでください。
2. ひな形ファイルのコピー先のディレクトリをサーチパスとして環境変数 PATH 又は path に設定します。
3. 手順 1 でコピーしたファイルの名称を HiRDB のコマンドの別名として使用したいコマンド名にします。例えば，データベース構成変更ユティリティ（pdmod）のコマンド名を変更したい場合，「hirmod」のように変更します。

4. コピーしたひな形ファイルを開き，次の図に示す「cc....cc」の部分を別名で実行させたい HiRDB のコマンドの名称に変更します。

**図 22-2 バッチファイルのひな形ファイル**

```
@echo off
setlocal

rem#set HiRDB command here
set HIRDB_COMMAND=cc....cc ← HiRDBのコマンドを設定する箇所（例：pdmodをhirmodに変更する
                                                          場合，pdmodを設定します）
set PARAM=
:GETPARAM
if"%1"==""" goto EXEC
  set PARAM=%PARAM% %1
  shift
goto GETPARAM
:EXEC
%PDDIR%¥bin¥%HIRDB_COMMAND%%PARAM%
endlocal
```

以上の設定によって，HiRDB のコマンドを任意の名称で実行できます。また，オプションも HiRDB のコマンドと同様に指定できます。

**（c）注意**

1. ひな形ファイルのコピー後の名称には，HiRDB のコマンドとは異なる名称を指定してください。
2. HiRDB 運用ディレクトリ下の二つのディレクトリ（%PDDIR%¥bin 及び %PDDIR%¥lib）は，アンインストール時にディレクトリごと削除される可能性があります。このため，この二つのディレクトリ以下にひな形ファイルをコピーしないでください。
3. バッチファイルのひな形ファイルをほかのバッチファイル中から呼び出す場合は，「call」を使用してください。「call」を使用しない場合は，呼び出し先から戻れなくなります。
4. ひな形ファイルの内容は，HiRDB のコマンド名の設定箇所以外は変更しないでください。
5. 作成した別名コマンドを実行中に，そのコマンド処理を中断させる場合には別名プロセスの延長で起動している HiRDB コマンドプロセスを停止させてください。別名プロセスを停止させるだけでは，HiRDB コマンドプロセスは停止しません。
6. 作成した別名コマンドを実行して，HiRDB コマンドが応答入力待ち状態のときに別名プロセスを停止させると，HiRDB コマンドの実行がエラーになったり応答入力待ち状態を継続している場合があります。応答入力待ちが継続されていた場合には，HiRDB コマンドプロセスを停止させてください。

## 22.3.3 サンプルで使用する HiRDB ファイルシステム領域名とユーザ作成ファイル名

サンプルで使用する HiRDB ファイルシステム領域名とサイズ及びユーザ作成ファイル名について説明します。

なお，ここで示す内容は，サンプルデータベースで使用している名称を示したもので，この名称と同じにしなければならないというものではありません。

### （1）HiRDB ファイルシステム領域名とサイズ

サンプルで使用する HiRDB ファイルシステム領域名とサイズを次の表に示します。

表 22-9 SPsetup.bat を実行して作成される HiRDB ファイルシステム領域名とサイズ

| HiRDB ファイルシステム領域の種類 | HiRDB ファイルシステム領域のサイズ（単位：メガバイト） | HiRDB ファイルシステム領域名 |
|---|---|---|
| システムファイル用 | 小規模→ 74<br>中規模→ 148<br>大規模→ 296 | rdsys011 ※1<br>rdsys012 ※1<br>rdsys013 ※1<br>rdsys014 ※1<br>rdsys015 ※1<br>rdsys016 ※1 |
| RD エリア用（システム用 RD エリア） | 小規模→ 20<br>中規模→ 40<br>大規模→ 80 | rdsys02 ※2 |
| 作業表用ファイル用 | － | rdsys03 ※2 |
| RD エリア用（ユーザ用 RD エリア） | 小規模→ 40<br>中規模→ 80<br>大規模→ 160 | rdsys04 ※2 |
| RD エリア用（ユーザ LOB 用 RD エリア） | 40 | rdsys05 ※2 |

注※ 1

SPsetup.bat 実行時に表示されるセットアップ内容の確認画面の［HiRDB システムファイル領域用ディレクトリ］で指定したディレクトリ下に作成されます。省略値は %PDDIR%¥area ディレクトリになります。

注※ 2

SPsetup.bat 実行時に表示されるセットアップ内容の確認画面の［HiRDB RD エリア領域用ディレクトリ］で指定したディレクトリ下に作成されます。省略値は %PDDIR%¥area ディレクトリになります。

## (2) ユーザ作成ファイル名

サンプルで使用するユーザ作成ファイル名を次の表に示します。

表 22-10 SPsetup.bat を実行して作成されるファイルの名称

| ファイルの種類 | | ファイル名 | 備考 |
|---|---|---|---|
| HiRDB システム定義ファイル | システム共通定義ファイル | %PDDIR%¥conf¥pdsys | ディレクトリはインストール時に作成されます。 |
| | ユニット制御情報定義ファイル | %PDDIR%¥conf¥pdutsys | |
| | シングルサーバ定義ファイル | %PDDIR%¥conf¥sds01 | |
| システムファイル | システムログファイル | rdsys011¥log1 ※1<br>rdsys012¥log2 ※1<br>rdsys013¥log3 ※1<br>rdsys014¥log4 ※1<br>rdsys015¥log5 ※1<br>rdsys016¥log6 ※1 | 6 グループ |

| ファイルの種類 | | ファイル名 | 備考 |
|---|---|---|---|
| | シンクポイントダンプファイル | rdsys014¥spd1 ※1<br>rdsys015¥spd2 ※1<br>rdsys016¥spd3 ※1 | 3 グループ |
| | ユニット用ステータスファイル | rdsys011¥utsts1a ※1<br>rdsys012¥utsts1b ※1<br>rdsys013¥utsts2a ※1<br>rdsys014¥utsts2b ※1<br>rdsys015¥utsts3a ※1<br>rdsys016¥utsts3b ※1 | ユニットごとに<br>二重化× 3 個 |
| | サーバ用ステータスファイル | rdsys011¥sts1a ※1<br>rdsys012¥sts1b ※1<br>rdsys013¥sts2a ※1<br>rdsys014¥sts2b ※1<br>rdsys015¥sts3a ※1<br>rdsys016¥sts3b ※1 | サーバごとに<br>二重化× 3 個 |
| システム用 RD エリア | マスタディレクトリ用 RD エリア | rdsys02¥rdmast ※2 | RD エリア名は RDMAST |
| | データディレクトリ用 RD エリア | rdsys02¥rddirt ※2 | RD エリア名は RDDIRT |
| | データディクショナリ用 RD エリア | rdsys02¥rddict ※2 | RD エリア名は RDDICT |
| | データディクショナリ LOB 用 RD エリア（ソース格納用） | rdsys02¥rtn_src ※2 | RD エリア名は DIC_RTN_SRC |
| | データディクショナリ LOB 用 RD エリア（オブジェクト格納用） | rdsys02¥rtn_obj ※2 | RD エリア名は DIC_RTN_OBJ |
| 作業表用ファイル | | rdsys03 ※2 | ― |
| RPC トレースファイル | | %PDDIR%¥spool¥pdrpctr | ― |
| ユーザ用 RD エリア（データ格納用） | | rdsys04¥rddata10 ※2 | RD エリア名は RDDATA10 |
| ユーザ用 RD エリア（インデクス格納用） | | rdsys04¥rdindx10 ※2 | RD エリア名は RDINDX10 |
| ユーザ LOB 用 RD エリア | | rdsys05¥rlob1 ※2 | RD エリア名は RLOB1 |
| | | rdsys05¥rlob2 ※2 | RD エリア名は RLOB2 |

（凡例）―：該当しません。

注※ 1

SPsetup.bat 実行時に表示されるセットアップ内容の確認画面の［HiRDB システムファイル領域用ディレクトリ］で指定したディレクトリ下に作成されます。省略値は %PDDIR%¥area ディレクトリになります。

注※ 2

SPsetup.bat 実行時に表示されるセットアップ内容の確認画面の［HiRDB RD エリア領域用ディレクトリ］で指定したディレクトリ下に作成されます。省略値は %PDDIR%¥area ディレクトリになります。

# 23 HiRDB サーバと HiRDB クライアント間の通信

この章では，HiRDB サーバと HiRDB クライアントの接続方法，DNS サーバやファイアウォールがある場合の設定などについて説明します。


23.1 HiRDB サーバと HiRDB クライアントの接続方法

23.2 DNS サーバで IP アドレスを管理する場合の設定

23.3 ファイアウォールや NAT が設置されている場合の設定

23.4 HiRDB が使用するポート数

23.5 HiRDB で指定するポート番号

23.6 HiRDB 予約ポート機能

# 23.1　HiRDB サーバと HiRDB クライアントの接続方法

HiRDB クライアントが HiRDB サーバに接続するには，クライアント環境定義の次に示すオペランドで**ホスト名**（又は IP **アドレス**）を指定する必要があります。

- PDHOST
- PDFESHOST

これらのオペランドに指定するホスト名は，システム共通定義の pdunit オペランドで指定したホスト名を指定します。

ただし，ネットワークの構成によっては pdunit オペランドで指定したホスト名を指定しても接続できない場合があります。DNS を使用している環境では「23.1.1　FQDN を指定した HiRDB サーバへの接続方法」を，HiRDB のサーバ間で使用しているネットワークと HiRDB クライアントと HiRDB サーバ間で使用しているネットワークが異なる場合は「23.1.2　マルチコネクションアドレス機能を使用した HiRDB サーバへの接続方法」を参照してください。

## 23.1.1　FQDN を指定した HiRDB サーバへの接続方法

pdunit オペランドで指定したホスト名は，HiRDB サーバにアクセスするすべてのクライアントマシンの hosts ファイルに IP アドレスとともに登録する必要があります。しかし，DNS を使用すると，hosts ファイルに登録する必要がなくなるため，登録処理や IP アドレス変更に伴う hosts ファイルの変更処理が不要になります。

ドメイン内に含まれるホスト上で稼働する HiRDB サーバと接続する場合は，PDHOST，PDFESHOST にサーバマシンの FQDN を指定することで，HiRDB サーバに接続できます。

クライアント環境定義に指定できる名称を次の表に示します。

**表 23-1　クライアント環境定義に指定できる名称**

| クライアント環境定義に指定する名称 | バージョン 05-02 以前 | バージョン 05-03 以降 |
|---|---|---|
| ホスト名 | ○ | ○ |
| FQDN | × | ○※ |

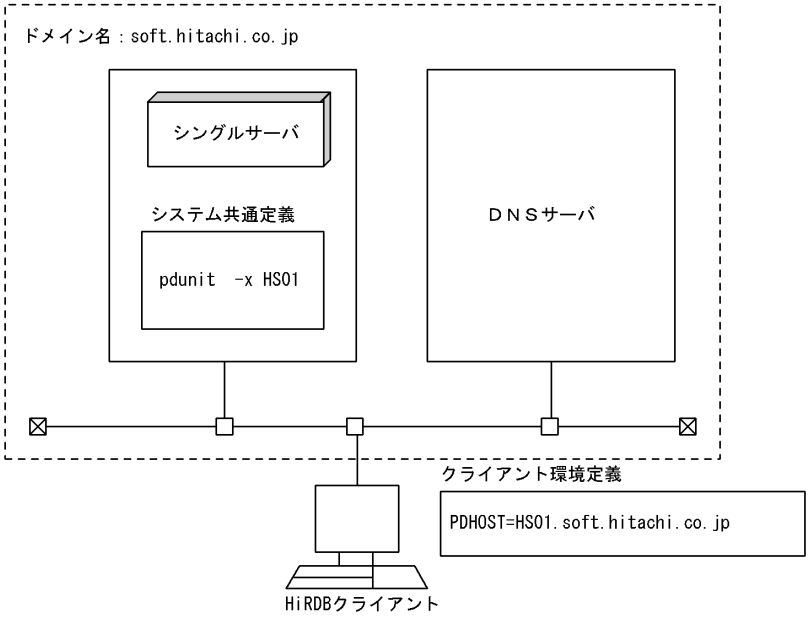
（凡例）
○：指定できます。
×：指定できません。

注※
大規模なネットワーク環境で，ホスト名や IP アドレスの登録，IP アドレスの変更に伴って hosts ファイルを変更したくない場合に使用します。

### （1）FQDN を指定して HiRDB サーバへ接続する場合のネットワーク構成例と定義例

FQDN を指定して HiRDB サーバへ接続する場合のネットワーク構成例と定義例を次の図に示します。

**図 23-1　FQDN を指定して HiRDB サーバへ接続する場合のネットワーク構成例と定義例**




〔説明〕

- システム共通定義の pdunit オペランドの -x オプションには，HiRDB サーバで使用するネットワークのホスト名（HS01）を指定します。
- クライアント環境定義の PDHOST には，HiRDB サーバの FQDN（HS01.soft.hitachi.co.jp）を指定します。

### (2) 注意事項

1. バージョン 05-03 より前の HiRDB サーバに接続する場合は，クライアント環境定義の PDHOST，PDFESHOST に FQDN を指定できません。指定するとクライアント最大待ち時間（PDCWAITTIME 指定値）経過後，キャンセル処理できないでサーバプロセスが残ることがあります。
2. HiRDB サーバで定義するホスト名に FQDN を指定できません。
3. HiRDB サーバが使用するネットワークと HiRDB クライアントが接続するネットワークが異なる場合は，マルチコネクションアドレス機能を使用して，HiRDB サーバに接続してください。マルチコネクションアドレス機能については，「23.1.2　マルチコネクションアドレス機能を使用した HiRDB サーバへの接続方法」を参照してください。

## 23.1.2　マルチコネクションアドレス機能を使用した HiRDB サーバへの接続方法

ネットワークの構成によっては pdunit オペランドで指定したホスト名を指定しても，HiRDB サーバと接続できないことがあります。例えば，HiRDB クライアントと HiRDB サーバ間で使用しているネットワークと，HiRDB サーバのサーバマシン間で使用しているネットワークが異なる場合がこれに該当します。

このような場合，**マルチコネクションアドレス機能**を使用します。この機能を使用すると，PDHOST 又は PDFESHOST オペランドに，pdunit オペランドと同じホスト名を指定しなくても，HiRDB サーバと接続できるようになります。

## (1) マルチコネクションアドレス機能の使用方法

マルチコネクションアドレス機能を使用するには，システム共通定義の pdstart オペランドで -m オプションを指定します。なお，系切り替え機能を使用する場合は，-m オプションのほかに -n オプションも指定します。

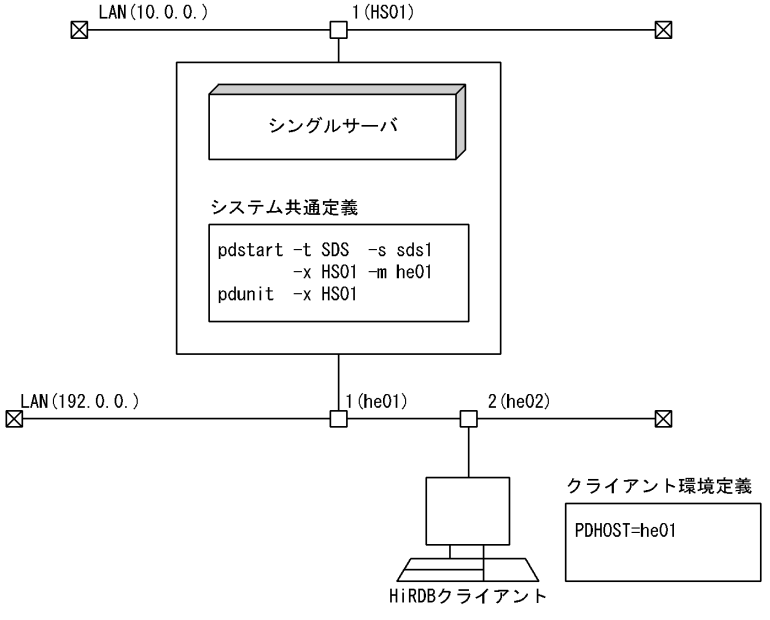
-m 及び -n オプションには，HiRDB クライアントがネットワークを経由して接続できる HiRDB サーバのホスト名を指定します。pdunit オペランドに指定したホスト名と同じ必要はありません。

## (2) マルチコネクションアドレス機能を使用したネットワーク構成例と定義例

### (a) HiRDB/ シングルサーバの場合

マルチコネクションアドレス機能を使用したネットワーク構成例と定義例（HiRDB/ シングルサーバの場合）を次の図に示します。

**図 23-2 ネットワーク構成例と定義例（HiRDB/ シングルサーバの場合）**



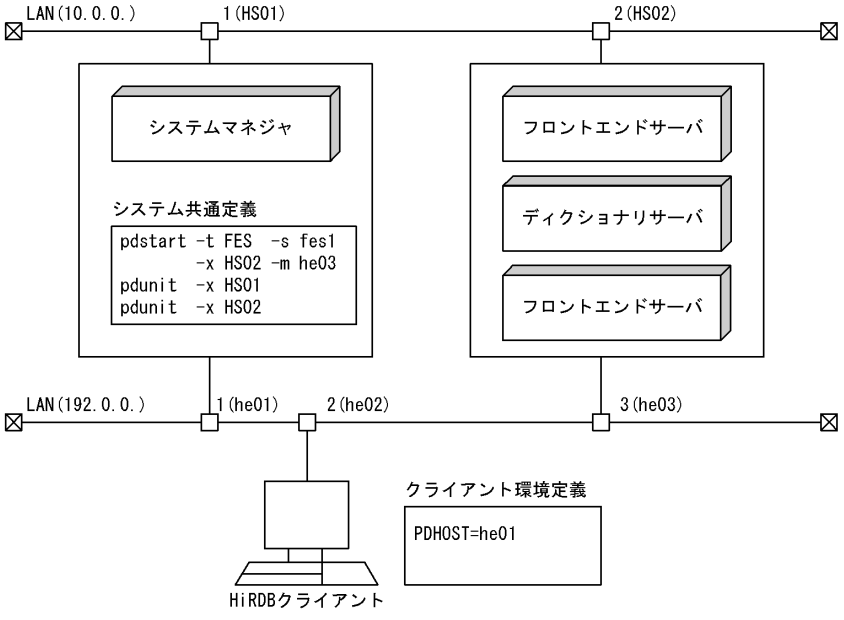

〔説明〕

- pdunit オペランドの -x オプションには，HiRDB サーバ間で使用するネットワークのホスト名（HS01）を指定します。
- pdstart オペランドの -m オプションには，HiRDB クライアントと HiRDB/ シングルサーバ間で使用するネットワークのホスト名（he01）を指定します。
- クライアント環境定義の PDHOST オペランドには，HiRDB クライアントと HiRDB/ シングルサーバ間で使用するネットワークのホスト名（he01）を指定します。

### (b) HiRDB/ パラレルサーバの場合

マルチコネクションアドレス機能を使用したネットワーク構成例と定義例（HiRDB/ パラレルサーバの場合）を次の図に示します。

**図 23-3　ネットワーク構成例と定義例（HiRDB/ パラレルサーバの場合）**



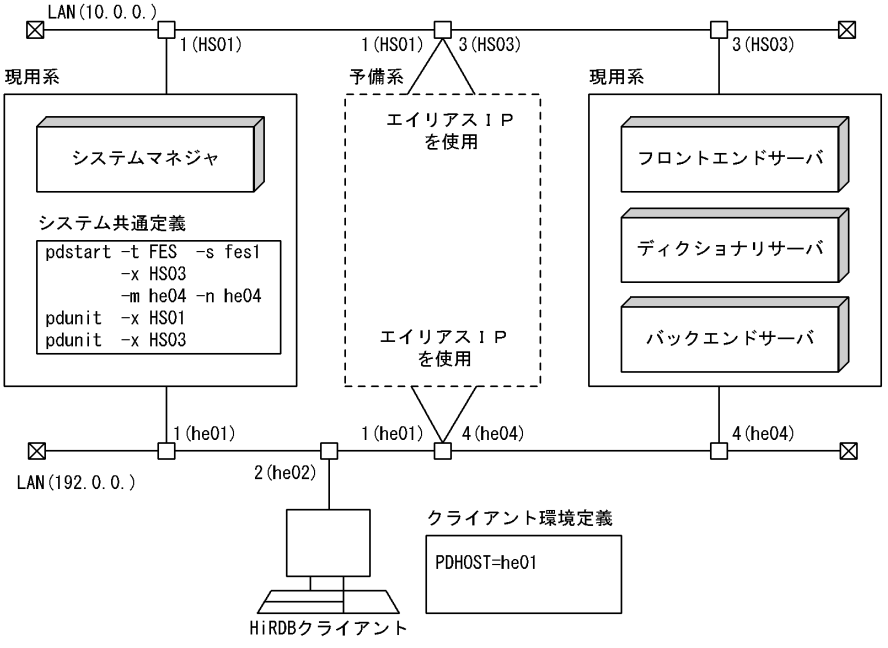

〔説明〕

- pdunit オペランドの -x オプションには，HiRDB サーバ間で使用するネットワークのホスト名（HS01，HS02）を指定します。
- pdstart オペランド（フロントエンドサーバを定義する pdstart オペランド）の -m オプションには，HiRDB クライアントと HiRDB サーバ間で使用するネットワークのホスト名（he03）を指定します。
- クライアント環境定義の PDHOST オペランドには，HiRDB クライアントと HiRDB サーバ間で使用するネットワークのホスト名（システムマネジャがあるホスト名：he01）を指定します。

### (c) HiRDB/ パラレルサーバの場合（IP アドレスを引き継ぐ系切り替えをする場合）

マルチコネクションアドレス機能を使用したネットワーク構成と定義例（IP アドレスを引き継ぐ系切り替えをする場合）を次の図に示します。

**図 23-4 ネットワーク構成例と定義例（IP アドレスを引き継ぐ系切り替えをする場合）**



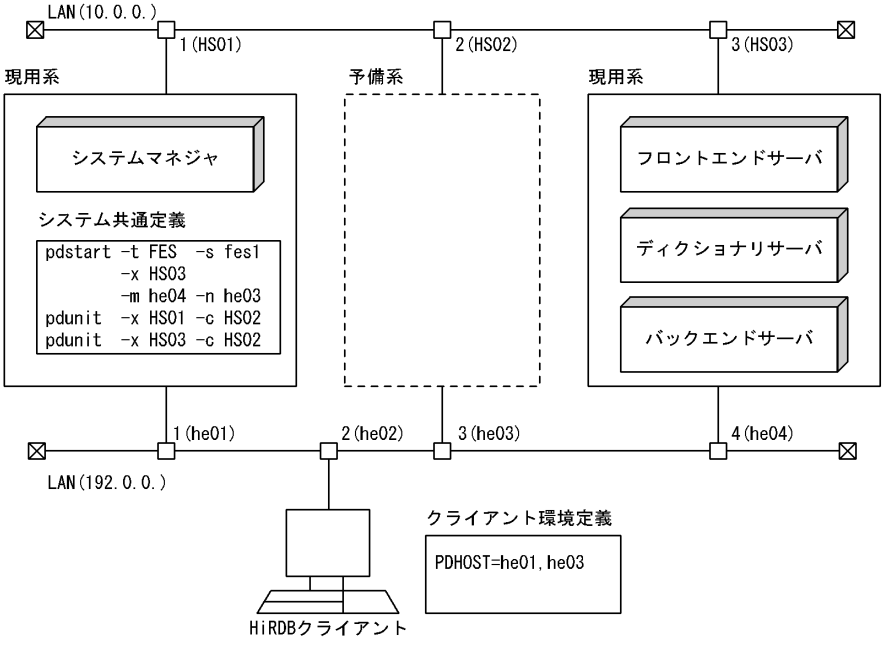

〔説明〕

- pdunit オペランドの -x オプションには，HiRDB サーバ間で使用するネットワークのホスト名（HS01，HS03）を指定します。
- pdstart オペランド（フロントエンドサーバを定義する pdstart オペランド）の -m オプションには，HiRDB クライアントと HiRDB サーバ間で使用するネットワークのホスト名（he04）を指定します。-n オプションには，予備系のホスト名（he04）を指定します。
- クライアント環境定義の PDHOST オペランドには，HiRDB クライアントと HiRDB サーバ間で使用するネットワークのホスト名（システムマネジャがあるホスト名：he01）を指定します。

### (d) HiRDB/ パラレルサーバの場合（IP アドレスを引き継がない系切り替えをする場合）

マルチコネクションアドレス機能を使用したネットワーク構成例と定義例（IP アドレスを引き継がない系切り替えをする場合）を次の図に示します。

**図 23-5　ネットワーク構成例と定義例（IP アドレスを引き継がない系切り替えをする場合）**




〔説明〕

- pdunit オペランドの -x オプションには，HiRDB サーバ間で使用するネットワークのホスト名（HS01，HS03）を指定します。-c オプションには，予備系のホスト名（HS02）を指定します。
- pdstart オペランド（フロントエンドサーバを定義する pdstart オペランド）の -m オプションには，HiRDB クライアントと HiRDB サーバ間で使用するネットワークのホスト名（he04）を指定します。-n オプションには，予備系のホスト名（he03）を指定します。
- クライアント環境定義の PDHOST オペランドには，HiRDB クライアントと HiRDB サーバ間で使用するネットワークのホスト名（システムマネジャがあるホスト名：he01）を指定します。また，予備系のホスト名（he03）も指定します。

# 23.2 DNS サーバで IP アドレスを管理する場合の設定

DNS サーバで IP アドレスを管理する場合の HiRDB システムには次の 2 とおりがあります。

- サーバマシンが同一ドメイン内に存在する場合
- サーバマシンが複数のドメインにわたって存在する場合

それぞれの HiRDB の設定方法を説明します。

## (1) 同一ドメイン内での HiRDB の設定方法

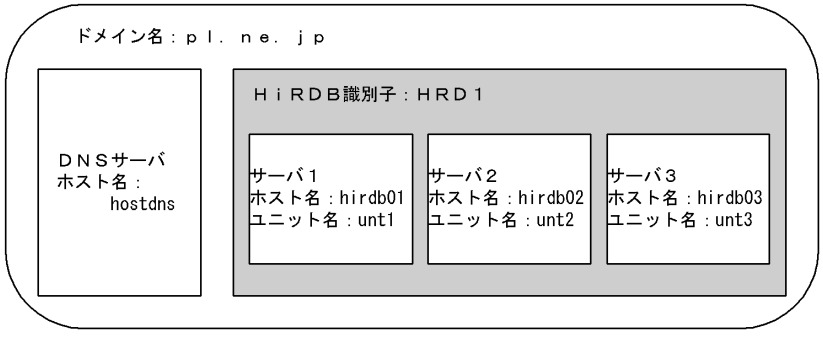
サーバマシンが同一ドメイン内に存在する場合，pdunit オペランド及び pdstart オペランドのホスト名称に「ホスト名」又は「FQDN（完全修飾子付きドメイン名称。ただし 32 文字以内）」のどちらかを指定します。これによって，DNS サーバで IP アドレスが管理でき，hosts ファイルの設定が不要になります。

具体的には，次のオプションでホスト名か FQDN を指定します。

- pdunit オペランド：-x 及び -c オプション
- pdstart オペランド：-x，-m，及び -n オプション

同一ドメインのシステム構成例を次の図に示します。

**図 23-6 同一ドメインのシステム構成例**




この場合の pdunit -x の指定例を次に示します。

● ホスト名指定の場合
pdunit -x hirdb01 -u unt1 -d " 運用ディレクトリ名 " -p ポート番号 …
pdunit -x hirdb02 -u unt2 -d " 運用ディレクトリ名 " -p ポート番号 …
pdunit -x hirdb03 -u unt3 -d " 運用ディレクトリ名 " -p ポート番号 …

● FQDN 指定の場合
pdunit -x hirdb01.p1.ne.jp -u unt1 -d " 運用ディレクトリ名 " -p ポート番号 …
pdunit -x hirdb02.p1.ne.jp -u unt2 -d " 運用ディレクトリ名 " -p ポート番号 …
pdunit -x hirdb03.p1.ne.jp -u unt3 -d " 運用ディレクトリ名 " -p ポート番号 …

## (2) 複数ドメインでの HiRDB の設定方法

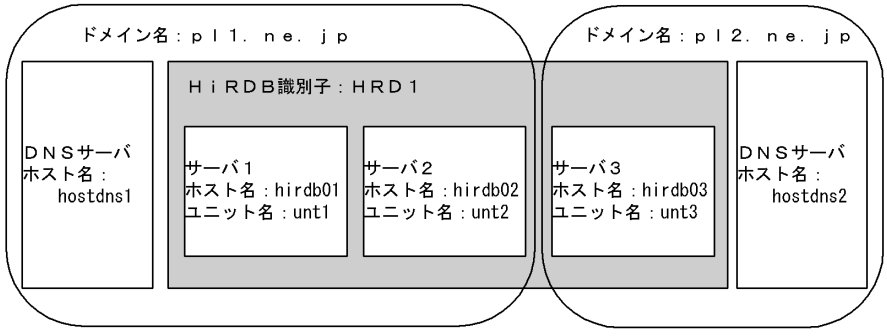
サーバマシンが複数のドメインにわたって存在する場合，pdunit オペランド及び pdstart オペランドのホスト名称に「FQDN（完全修飾子付きドメイン名称：ただし 32 文字以内）」を指定します。これによって，DNS サーバで IP アドレスが管理でき，hosts ファイルの設定が不要になります。

具体的には，次のオプションで FQDN を指定します。

- pdunit オペランド：-x，及び -c オプション
- pdstart オペランド：-x，-m，及び -n オプション

複数ドメインのシステム構成例を次の図に示します。

**図 23-7　複数ドメインのシステム構成例**




この場合の pdunit -x の指定例を次に示します。

● FQDN 指定の場合

pdunit -x hirdb01.pl1.ne.jp -u unt1 -d " 運用ディレクトリ名 " -p ポート番号 …
pdunit -x hirdb02.pl1.ne.jp -u unt2 -d " 運用ディレクトリ名 " -p ポート番号 …
pdunit -x hirdb03.pl2.ne.jp -u unt3 -d " 運用ディレクトリ名 " -p ポート番号 …

# 23.3 ファイアウォールや NAT が設置されている場合の設定

HiRDB サーバと HiRDB クライアント間に，ファイアウォールや NAT が設置されている場合の HiRDB の環境設定について説明します。

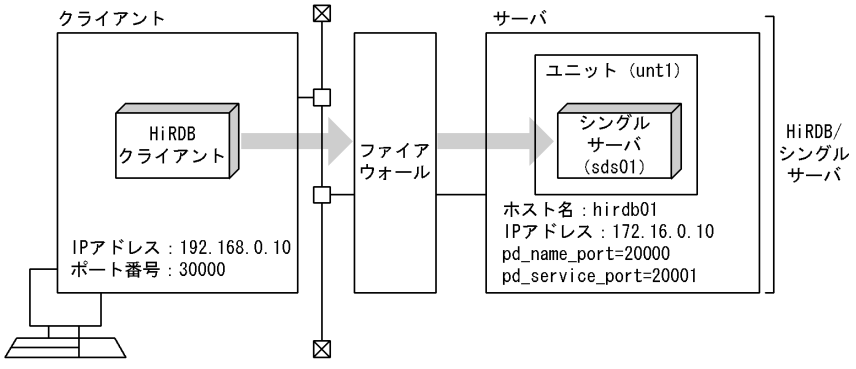
## 23.3.1 HiRDB/ シングルサーバ側にファイアウォールを設置した場合

次の図のように HiRDB/ シングルサーバ側にファイアウォールが設置され，そのファイアウォールが次のように設定されているとします。

**ファイアウォールの設定**

- 方向：受信
- 透過させる IP アドレス：172.16.0.10
- 透過させるポート番号：20000，20001

**図 23-8 ファイアウォールが HiRDB/ シングルサーバ側に設置されているネットワーク構成例**




この場合，サーバマシン及びクライアントマシンの設定は次のようになります。ファイアウォールを設置する場合，次のどれかのオペランドを指定する必要があります。

- pd_service_port オペランド
- pd_scd_port オペランド
- pdunit オペランドの -s オプション

ファイアウォールだけ設置する場合，クライアント環境定義（PDSERVICEPORT オペランド）を指定する必要はありません。

**サーバマシンの設定**

- システム共通定義ファイル
  set pd_name_port= 20000
  set pd_service_port= 20001
  pdunit -x hirdb01 -u unt1
  pdstart -t SDS -s sds01 -u unt1

**クライアントマシンの設定**

- クライアント環境定義

PDHOST　hirdb01
PDNAMEPORT　20000
PDCLTRCVPORT　30000※

- hosts ファイル
  172.16.0.10　hirdb01

注※　クライアント側にファイアウォールがある場合に指定してください。

## 23.3.2　HiRDB/ シングルサーバ側にファイアウォールと NAT を設置した場合

次の図のように HiRDB/ シングルサーバ側にファイアウォールと NAT が設置され，それらが次のように設定されているとします。

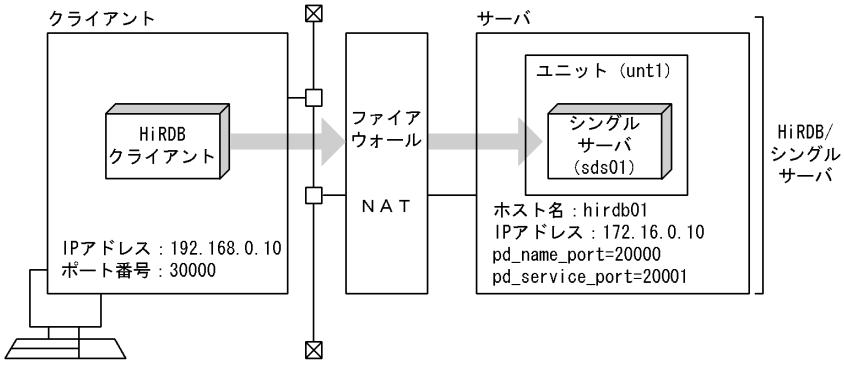
**ファイアウォールの設定**

- 方向：受信
- 透過させる IP アドレス：172.16.0.10
- 透過させるポート番号：20000，20001

**NAT によるアドレス変換**

128.1.1.1 ←→ 172.16.0.10
なお，NAPT（IP マスカレード）のような，グローバル IP アドレスとローカル IP アドレスを 1 対複数で変換するような機能には対応していません。1 対 1 の変換だけサポートしています。

**図 23-9　ファイアウォールと NAT が HiRDB/ シングルサーバ側に設置されているネットワーク構成例**




この場合，高速接続機能を使用する設定にしてください。サーバマシン及びクライアントマシンの設定は次のようになります。

**サーバマシンの設定**

- システム共通定義ファイル
  set pd_name_port = 20000
  set pd_service_port = 20001
  pdunit -x hirdb01 -u unt1
  pdstart -t SDS -s sds01 -u unt1

**クライアントマシンの設定**

- クライアント環境定義

PDHOST　hirdb01
PDNAMEPORT　20000
PDSERVICEGRP　sds01
PDSERVICEPORT　20001
PDSRVTYPE　PC
PDCLTRCVPORT　30000※

- hosts ファイル
  128.1.1.1　hirdb01

注※　クライアント側にファイアウォールがある場合に指定してください。

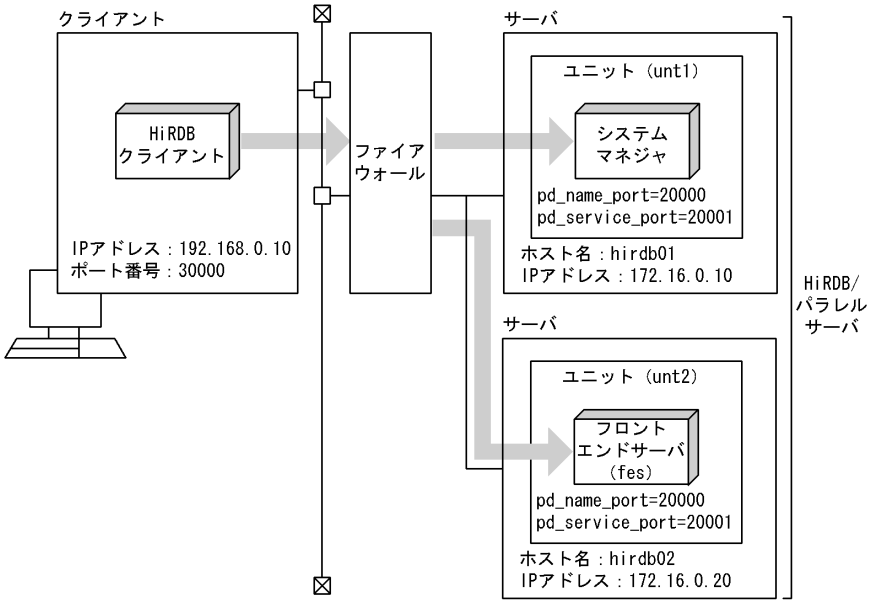
## 23.3.3　HiRDB/ パラレルサーバ側にファイアウォールを設置した場合

次の図のように HiRDB/ パラレルサーバ側にファイアウォールが設置され，そのファイアウォールが次のように設定されているとします。

**ファイアウォールの設定**

- 方向：受信
- 透過させる IP アドレス：172.16.0.10，172.16.0.20
- 透過させるポート番号：20000，20001

**図 23-10　ファイアウォールが HiRDB/ パラレルサーバ側に設置されているネットワーク構成例**




この場合，サーバマシン及びクライアントマシンの設定は次のようになります。ファイアウォールを設置する場合，次のどれかのオペランドを指定する必要があります。

- pd_service_port オペランド
- pd_scd_port オペランド
- pdunit オペランドの -s オプション

ファイアウォールだけ設置する場合，クライアント環境定義（PDSERVICEPORT オペランド）を指定する必要はありません。

**サーバマシンの設定**

- システム共通定義ファイル
  set pd_name_port = 20000
  set pd_service_port = 20001
  pdunit -x hirdb01 -u unt1
  pdunit -x hirdb02 -u unt2
  pdstart -t MGR -u unt1
  pdstart -t FES -s fes -u unt2

**クライアントマシンの設定**

- クライアント環境定義
  PDHOST　hirdb01
  PDNAMEPORT　20000
  PDCLTRCVPORT　30000※
- hosts ファイル
  172.16.0.10　hirdb01
  172.16.0.20　hirdb02

注※　クライアント側にファイアウォールがある場合に指定してください。

### 23.3.4　HiRDB/ パラレルサーバ側にファイアウォールと NAT を設置した場合

次の図のように HiRDB/ パラレルサーバ側にファイアウォールと NAT が設置され，それらが次のように設定されているとします。

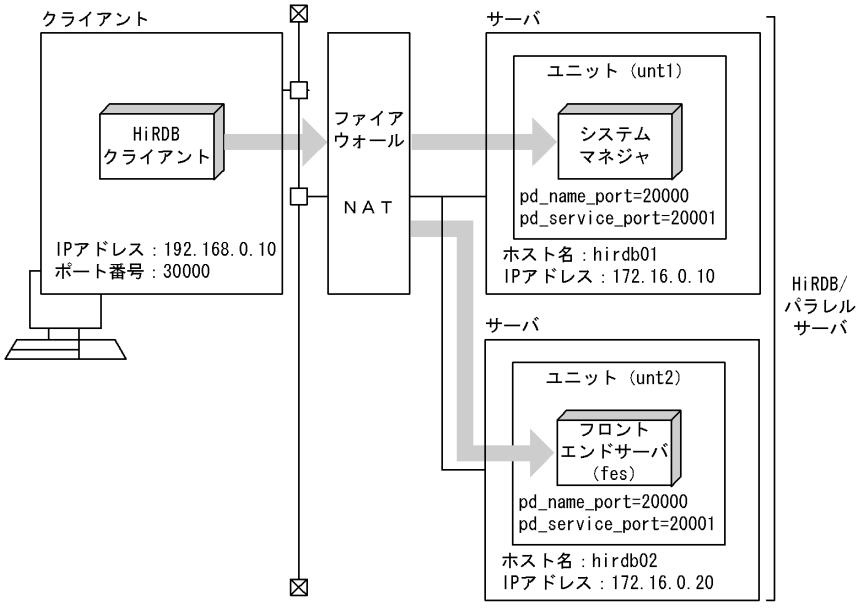
**ファイアウォールの設定**

- 方向：受信
- 透過させる IP アドレス：172.16.0.10，172.16.0.20
- ポート番号：20000，20001

**NAT によるアドレス変換**

128.1.1.1 ←→ 172.16.0.10
129.1.1.2 ←→ 172.16.0.20
なお，NAPT（IP マスカレード）のような，グローバル IP アドレスとローカル IP アドレスを 1 対複数で変換するような機能には対応していません。1 対 1 の変換だけサポートしています。

**図 23-11 ファイアウォールと NAT が HiRDB/ パラレルサーバ側に設置されているネットワーク構成例**




この場合，高速接続機能を使用する設定にしてください。サーバマシン及びクライアントマシンの設定は次のようになります。

**サーバマシンの設定**

- システム共通定義ファイル
  set pd_name_port = 20000
  set pd_service_port = 20001
  pdunit -x hirdb01 -u unt1
  pdunit -x hirdb02 -u unt2
  pdstart -t MGR -u unt1
  pdstart -t FES -s fes -u unt2

**クライアントマシンの設定**

- クライアント環境定義
  PDHOST　hirdb01
  PDNAMEPORT　20000
  PDSERVICEGRP　fes
  PDSERVICEPORT　20001
  PDFESHOST　hirdb02
  PDSRVTYPE　PC
  PDCLTRCVPORT　30000 ※
- hosts ファイル
  128.1.1.1　hirdb01
  128.1.1.2　hirdb02

注※　クライアント側にファイアウォールがある場合に指定してください。

# 23.4　HiRDB が使用するポート数

HiRDB の通信処理では，pd_registered_port オペランドの指定がない場合，OS が自動的に割り当てる通信ポート番号を使用します。使用する通信ポート数は，pd_max_users オペランドの値やバックエンドサーバ数の増加に伴って増加します。ポート数が不足すると，処理が中断したり，他プログラムの通信処理に影響を与えたりします。

なお，pd_registered_port オペランドで HiRDB が通信処理で使用するポート番号の範囲を指定（HiRDB 予約ポート機能）できます。HiRDB 予約ポート機能については，「23.6　HiRDB 予約ポート機能」を参照してください。

## 23.4.1　ユニットが使用する通信ポート数の見積もり

HiRDB のユニットが使用する通信ポート数の目安を次に示します。

### (1) シングルサーバの場合

**計算式**

pd_max_users の値× 2 ＋ 1000

### (2) パラレルサーバの場合：

マルチフロントエンドサーバ構成の場合は，フロントエンドサーバごとに f，F を決定し，算出したポート数の合算となります。フロントエンドサーバごとのポート数の目安を算出する計算式を次に示します。

**計算式**

{b ×［k ×〔(b － 1）÷ 2 ＋ B ＋ 1〕＋ 1］
＋ f ×（k ×（b ＋ B）＋ D ＋ 2）＋ d × 2}
× pd_max_users の値＋ 1000

b：ユニット内のバックエンドサーバ数

B：ユニット外のバックエンドサーバ数

f：ユニット内のフロントエンドサーバ数
フロントエンドサーバごとに 1，又は 0 のどちらかの値になります。
ユニット内にフロントエンドサーバがある場合：1
ユニット外にフロントエンドサーバがある場合：0

d：ユニット内のディクショナリサーバ数

D：ユニット外のディクショナリサーバ数

k：2 ÷ 3

## 23.4.2　注意事項

- 実行する SQL 文によっては，目安以上のポート数が必要となる場合があります。
- OS が自動的に割り当てるポート数で足りない場合は，pd_registered_port オペランドで指定し直してください。
- ポート番号は，HiRDB が解放しても OS がすぐに解放しない場合（TIME_WAIT 状態）があります。

そのため，目安以上のポート数を一時的に使用する場合があります。

- 次に示すシステムの場合は，高速接続機能を使用して UAP が HiRDB に接続するときに使用するポート数を少なくしてください。高速接続機能については，マニュアル「HiRDB Version 8 UAP 開発ガイド」を参照してください。
  - pd_max_users の値が 400 を超えるシステムの場合
  - システム内のバックエンドサーバ数が 32 を超えるシステムの場合
  - システム内のフロントエンドサーバ数が 32 を超えるシステムの場合
  - ユニット数が 64 を超えるシステムの場合
- システム内のバックエンドサーバ数が多いシステム構成の場合，計算した値が OS のポート番号の上限値や pd_registered_port に指定したポート番号を超えるおそれがあります。その場合の対策については，「9.5　大規模なシステムの構築」の「(2)　注意事項」を参照してください。また，バックエンドサーバ接続保持機能を使用してください。バックエンドサーバ接続保持機能については，マニュアル「HiRDB Version 8 システム運用ガイド」を参照してください。

### 23.4.3　計算例

● 1 ユニット構成の HiRDB/ パラレルサーバ（FES，DS，BES が 5 個）で，pd_max_users ＝ 1000 とした場合

```
｛5×［2÷3×（0＋0）+1］
＋1×（2÷3×0＋0＋2）＋1×（0＋1）｝
×1000＋1000＝9000
```

9000 個のポート数が目安となります。

● ユニット構成のパラレル（ユニット 1（FES，DS，BES が 2 個），ユニット 2（BES が 3 個））で，pd_max_users ＝ 900 とした場合

ユニット 1：

```
｛2×［2÷3×（3＋0）＋1］
＋1×（2÷3×3＋0＋2）＋1×（0＋1）｝
×900+1000＝10900
```

ユニット 2：

```
｛3×［2÷3×（2＋1）＋1］
＋0×（2÷3×2＋1＋2）＋0×（1＋1）｝
×900+1000＝9100
```

それぞれに 10900 個，9100 個のポート数が目安となります。

# 23.5　HiRDB で指定するポート番号

## 23.5.1　HiRDB で指定するポート番号の一覧

HiRDB で指定するポート番号を次の表に示します。

**表 23-2　HiRDB で指定するポート番号の一覧**

| 設定箇所 | 指定する環境変数又はオペランド | 説明 |
|---|---|---|
| クライアント環境定義 | PDNAMEPORT | HiRDB のポート番号 |
| | HiRDB_PDNAMEPORT | |
| | PDSERVICEPORT | 高速接続用のポート番号 |
| | PDFESHOST | フロントエンドサーバがあるユニットのポート番号 |
| | PDCLTRCVPORT | クライアントの受信ポート番号 |
| | PDASTPORT | HiRDB Control Manager - Agent のポート番号 |
| システム共通定義 | pd_name_port | HiRDB のポート番号 |
| | pdunit -p | |
| | pd_service_port | スケジューラプロセスのポート番号 |
| | pd_scd_port | |
| | pdunit -s | |
| | pd_trn_port | トランザクションサーバプロセスのポート番号 |
| | pdunit -t | |
| | pd_mlg_port | メッセージログサーバプロセスのポート番号 |
| | pdunit -m | |
| | pd_alv_port | ユニット監視プロセスのポート番号 |
| | pdunit -a | |
| | pd_registered_port | HiRDB 予約ポート機能で使用するポート番号 |
| ユニット制御情報定義 | pd_service_port | スケジューラプロセスのポート番号 |
| | pd_registered_port | HiRDB 予約ポート機能で使用するポート番号 |
| %windir%¥system32¥drivers¥etc¥services ファイル | ― | 連絡用ポート番号 |

## 23.5.2　ポート番号の指定方法

ここでは，ポート番号の指定方法について，それぞれ説明します。

### (1) クライアント環境定義

マニュアル「HiRDB Version 8 UAP 開発ガイド」の「クライアント環境定義（環境変数の設定）」を参照してください。

### (2) システム共通定義及びユニット制御情報定義

システム共通定義及びユニット制御情報定義のオペランドの指定有無と使用するポート番号について説明

します。

オペランドに指定したポート番号を固定したい場合，表中の「使用するポート番号」を参照して，そのポート番号を使用する組み合わせでオペランドを指定してください。例えば，「(a)HiRDB のポート番号」で，HiRDB/ パラレルサーバの場合，pd_name_port オペランドに指定したポート番号に固定したいとき，pdunit -p は指定を省略してください。

### (a) HiRDB のポート番号

HiRDB のポート番号は，pd_name_port オペランド及び pdunit オペランドの -p オプションに指定します。これらのオペランドの指定有無と使用するポート番号について説明します。

HiRDB/ シングルサーバの場合

| pd_name_port | pdunit -p | 使用するポート番号 |
|---|---|---|
| あり | あり | pd_name_port の指定 |
| | なし | |
| なし | あり | 20000 |
| | なし | |

HiRDB/ パラレルサーバの場合

| pd_name_port | pdunit -p | 使用するポート番号 |
|---|---|---|
| あり | あり | pdunit -p の指定 |
| | なし | pd_name_port の指定 |
| なし | あり | pdunit -p の指定 |
| | なし | 20000 |

### (b) スケジューラプロセスのポート番号

スケジューラプロセスのポート番号は，次のオペランドに指定します。

- pd_service_port オペランド
- pd_scd_port オペランド
- pdunit オペランドの -s オプション

これらのオペランドの指定有無と使用するポート番号を次に示します。

| 指定箇所 | | | | 使用するポート番号 |
|---|---|---|---|---|
| システム共通定義 | | | ユニット制御情報定義 | |
| pd_service_port | pd_scd_port | pdunit -s | pd_service_port | |
| あり | あり | あり | あり | pdunit -s の指定 |
| | | | なし | |
| | | なし | あり | pd_scd_port の指定 |
| | | | なし | |
| | なし | あり | あり | pdunit -s の指定 |
| | | | なし | |

| 指定箇所 | | | | 使用するポート番号 |
|---|---|---|---|---|
| システム共通定義 | | | ユニット制御情報定義 | |
| pd_service_port | pd_scd_port | pdunit -s | pd_service_port | |
| | | なし | あり | ユニット制御情報定義の pd_service_port の指定 |
| | | | なし | |
| なし | あり | あり | あり | pdunit -s の指定 |
| | | | なし | |
| | | なし | あり | pd_scd_port の指定 |
| | | | なし | |
| | なし | あり | あり | pdunit -s の指定 |
| | | | なし | |
| | | なし | あり | ユニット制御情報定義の pd_service_port の指定 |
| | | | なし | ※ |

注※

システム共通定義又はユニット制御情報定義に pd_registered_port オペランドを指定した場合，pd_registered_port オペランドに指定した範囲内のポート番号を使用します。pd_registered_port オペランドを指定しない場合，OS が自動的に割り当てたポート番号を使用します。

### (c) トランザクションサーバプロセス，メッセージログサーバプロセス，及びユニット監視プロセスのポート番号

トランザクションサーバプロセス，メッセージログサーバプロセス，及びユニット監視プロセスのポート番号は，次に示すオペランドに指定します。

- トランザクションサーバプロセス
  pd_tran_port オペランド及び pdunit オペランドの -t オプション
- メッセージログサーバプロセス
  pd_mlg_port オペランド及び pdunit オペランドの -m オプション
- ユニット監視プロセス
  pd_alv_port オペランド及び pdunit オペランドの -a オプション

これらのオペランドの指定有無と使用するポート番号を次に示します。

| pd_trn_port 又は pd_mlg_port 又は pd_alv_port | pdunit -t 又は -m 又は -a | 使用するポート番号 |
|---|---|---|
| あり | あり | pdunit -t，-m，又は -a の指定 |
| | なし | pd_trn_port 又は pd_mlg_port 又は pd_alv_port の指定 |
| なし | あり | pdunit -t，-m，又は -a の指定 |
| | なし | ※ |

注※

システム共通定義又はユニット制御情報定義に pd_registered_port オペランドを指定した場合，pd_registered_port オペランドに指定した範囲内のポート番号を使用します。pd_registered_port オ

ペランドを指定しない場合，OS が自動的に割り当てたポート番号を使用します。

#### (d) HiRDB 予約ポート機能で使用するポート番号

HiRDB 予約ポート機能で使用するポート番号は，pd_registered_port オペランドに指定します。このオペランドの指定有無と使用するポート番号を次に示します。

| 指定箇所 | | 使用するポート番号 |
|---|---|---|
| システム共通定義 | ユニット制御情報定義 | |
| pd_registered_port | pd_registered_port | |
| あり | あり | ユニット制御情報定義の指定 |
| | なし | システム共通定義の指定 |
| なし | あり | ユニット制御情報定義の指定 |
| | なし | ― |

（凡例）―：該当しません。

### (3) 連絡用ポート番号

「2.1.3　OS 環境ファイルの設定」を参照してください。

## 23.5.3　ポート番号の重複に関する注意事項

HiRDB システム定義（表 23-2 のシステム共通定義及びユニット制御情報定義）で指定するポート番号の重複についての注意事項を次に示します。

- 同じユニットでは，各オペランドに異なるポート番号を指定してください。
- HiRDB/ パラレルサーバで，一つのサーバマシン上で複数のユニットを起動する場合，それぞれのユニットの各オペランドには異なるポート番号を指定してください。
- 一つのサーバマシン上で，複数の HiRDB を起動する場合，それぞれの HiRDB の各オペランドには異なるポート番号を指定してください。
- 一つのサーバマシン上に HiRDB クライアントと HiRDB サーバがある場合，各オペランドには，クライアント環境定義の PDCLTRCVPORT と異なるポート番号を指定してください。
- 次の条件を満たすようにポート番号を指定してください。
  - ほかのプログラムプロダクトが使用しているポート番号と異なる
  - %windir%¥system32¥drivers¥etc¥services（DNS 環境の場合は定義した場所）に登録されたポート番号と異なる
  - OS が自動的に割り当てるポート番号※の範囲に含まれない
    注※　OS が自動的に割り当てるポート番号の範囲は OS によって異なります。
- HiRDB のポート番号の指定を省略した場合，HiRDB のポート番号に 20000 が仮定されます。そのため，各オペランドには 20000 を指定しないでください。

# 23.6　HiRDB 予約ポート機能

HiRDB 予約ポート機能とは，pd_registered_port オペランドでポート番号の範囲を指定して，特定の通信ポート番号の範囲で通信できるようにする機能です。この機能によって，次のようなことが防げます。

- HiRDB 以外のプログラムが OS の自動割り当てポート番号を多数使用した通信処理をしている場合，通信ポート番号不足が発生して処理が中断される
- HiRDB が大量の通信ポート番号を使用し，他プログラムの通信処理に影響を与える
  確保したポートは HiRDB が解放しても，ある一定時間は再利用されません。そのため，短時間にポートを使用する処理が大量に発生した場合，ポート番号不足が発生する可能性があります。

HiRDB 予約ポート機能は，次のような運用の場合に使用することを検討してください。

- HiRDB クライアントからの CONNECT 又は DISCONNECT が 1 分間に数千回発生する場合
- HiRDB/ パラレルサーバで，トランザクションが 1 分間に数千個発生する場合

なお，HiRDB 予約ポート機能を使用する場合の定義例，及び注意事項については，マニュアル「HiRDB Version 8 システム定義」の pd_registered_port オペランドの説明を参照してください。

## 23.6.1　HiRDB 予約ポート数の見積もり

### (1) 統計情報からの見積もり

統計情報から HiRDB の通信ポート番号の使用数を計れます。統計解析ユティリティの「システムの稼働に関する統計情報」で，次の情報から HiRDB の通信ポート数を算出します。なお，これらの詳細は，マニュアル「HiRDB Version 8 コマンドリファレンス」を参照してください。

- HiRDB 予約ポート使用数
- HiRDB 予約ポートオーバー時の OS 自動割り当てポート使用数

**計算式**

HiRDB 予約ポート使用数＋ HiRDB 予約ポートオーバー時の OS 自動割り当てポート使用数＋ 100 ※

注※　予備として加算しています。

### (2) 推奨値の見積もり

推奨値は，「23.4.1　ユニットが使用する通信ポート数の見積もり」の計算式で求められます。なお，この値は目安です。運用上は「(1) 統計情報からの見積もり」の値を使用してください。

# 付録


付録 A　HiRDB の最大値・最小値

付録 B　HiRDB のプロセス一覧

付録 C　Q & A

付録 D　バッチファイルによる環境設定

# 付録 A　HiRDB の最大値・最小値

## 付録 A.1　システム構成に関する最大値と最小値

HiRDB のシステム構成に関する最大値と最小値を次の表に示します。

**表 A-1　HiRDB のシステム構成に関する最大値と最小値**

| 項　目 | 最小値 | 最大値 | 単位 |
|---|---|---|---|
| 1 ユニットに作成できるサーバ数 | 1 | 34 | 個 |
| シングルサーバ数 | 1 | 1 | 個 |
| システムマネジャ数 | 1 | 1 | 個 |
| フロントエンドサーバ数 | 1 | 1,024 | 個 |
| ディクショナリサーバ数 | 1 | 1 | 個 |
| バックエンドサーバ数 | 1 | 16,382 | 個 |
| 総 RD エリア数 | 3 | 8,388,592 | 個 |
| マスタディレクトリ用 RD エリア数 | 1 | 1 | 個 |
| データディレクトリ用 RD エリア数 | 1 | 1 | 個 |
| データディクショナリ用 RD エリア数 | 1 | 59 | 個 |
| 解析情報表及び運用履歴表を格納するデータディクショナリ用 RD エリア数 | 1 | 1 | 個 |
| ユーザ用 RD エリア数 | 0 | 8,388,589 | 個 |
| データディクショナリ LOB 用 RD エリア数 | 0 | 2 | 個 |
| ユーザ LOB 用 RD エリア数 | 0 | 8,388,325 | 個 |
| レジストリ用 RD エリア数 | 0 | 1 | 個 |
| レジストリ LOB 用 RD エリア数 | 0 | 1 | 個 |
| リスト用 RD エリア | 0 | 8,388,588 | 個 |
| 1RD エリア中の HiRDB ファイル数 | 1 | 16 | 個 |
| 1RD エリア中の実表数と順序数生成子数の合計 | 0 | 500 | 個 |
| 1RD エリア中のインデクス数 | 0 | 500 | 個 |
| 1RD エリア中のリスト数 | 0 | 50,000 | 個 |
| 総 HiRDB ファイル数 | 1 | 134,217,728 | 個 |
| 同時アクセス可能実表数 | 4 | 32,000 | 個 |
| 最大同時接続数 | 1 | HiRDB/ シングルサーバの場合：3,000<br>HiRDB/ パラレルサーバの場合：2,000 ※1 | 個 |
| HiRDB で作成可能なユーザ数※2 | 1 | 無制限 | 個 |
| 同時リスト所有可能ユーザ数 | 0 | 32,767 | 個 |
| 1 ユーザ当たりのリスト作成数 | 0 | 32,767 | 個 |
| 1 トランザクション当たりの作業表作成数 | 0 | 255 | 個 |

| 項　目 | 最小値 | 最大値 | 単位 |
|---|---|---|---|
| 1 サーバ当たりのグローバルバッファ数※3 | 1 | 2,000,000 | 個 |
| HiRDB ファイルシステム領域長 | 1 | 1,048,575 | メガバイト |
| Windows のファイル容量 | 0 | $2^{63}-1$ | バイト |

注※ 1

マルチフロントエンドサーバ構成の場合，フロントエンドサーバ数× pd_max_users 数の上限が 2,000 になります。

注※ 2

- ディレクトリサーバ連携機能を使用していない場合
  1 ユーザ当たりディクショナリ表（SQL_USERS）を 1 行消費するため，データディクショナリ用 RD エリアの容量に依存します。
- ディレクトリサーバ連携機能を使用している場合
  CONNECT 権限だけを持つユーザについては，ディレクトリサーバの仕様に依存します。
  CONNECT 権限以外の権限を持つユーザについては，ディレクトリサーバ連携機能を使用していない場合と同様です。

注※ 3

ただし，システム全体では 2,147,483,647 が上限となります。

## 付録 A.2　データベースに関する最大値と最小値

データベースに関する最大値と最小値を次の表に示します。

**表 A-2　データベースに関する最大値と最小値**

| 項　目 | | 最小値 | 最大値 | 単位 |
|---|---|---|---|---|
| 文字データの長さ | CHAR | 1 | 30,000 | 字（バイト） |
| （定義長） | VARCHAR | 1 | 32,000 | 字（バイト） |
| 各国文字データの長さ | NCHAR | 1 | 15,000 | 字 |
| （定義長） | NVARCHAR | 1 | 16,000 | 字 |
| 混在文字データの長さ | MCHAR | 1 | 30,000 | バイト |
| （定義長） | MVARCHAR | 1 | 32,000 | バイト |
| パック形式 10 進数の精度 | DECIMAL 又は NUMERIC | 1 | 38 | けた |
| パック形式 10 進数の位取り | DECIMAL 又は NUMERIC | 0 | 38 | けた |
| 時刻印データの小数秒精度 | TIMESTAMP | 0 | 6 | けた |
| BLOB データの長さ | | 1 | 2,147,483,647 | バイト |
| BINARY データの長さ（定義長） | | 1 | 2,147,483,647 | バイト |
| 表中の列数 | | 1 | 30,000 | 列 |
| 表中のインデクス数 | | 0 | 255 | 個 |
| インデクスの構成列数 | | 1 | 16 | 列 |
| クラスタキーの構成列数 | | 1 | 16 | 列 |
| 表を分割格納する RD エリア数 | | 1 | 1,024 | 個 |

| 項　目 | 最小値 | 最大値 | 単位 |
|---|---|---|---|
| 表を分割配置する BES 数 | 1 | 1,024 | 個 |
| 横分割表を定義時の格納条件に指定する定数の総数（格納条件を省略した場合も 1 と数える） | 1 | 15,000 | 個 |
| ビュー表の基になる表数 | 1 | 64 | 個 |
| ビュー表の列数 | 1 | 30,000 | 列 |
| 主キーの構成列数 | 1 | 16 | 列 |
| 外部キーの構成列数 | 1 | 16 | 列 |
| 1 表中の外部キー数 | 0 | 255 | 個 |
| 一つの主キーを参照する外部キー数 | 0 | 255 | 個 |
| 1 表中に指定できる検査制約定義数 | 0 | 254 | 個 |
| 1 表中に定義できる検査制約中で指定した論理演算子 AND，OR，及び検査制約定義の数の合計（CASE 式の WHEN 探索条件中の論理演算子の AND 及び OR は除きます） | 0 | 254 | 個 |
| 識別子の長さ<br>（表識別子，列名，データ型識別子，インデクス型識別子，属性名，ルーチン識別子，相関名，インデクス識別子，カーソル名，SQL 文識別子，RD エリア名，埋込み変数名，標識変数名，パスワード，制約名，条件名，SQL 変数名，問合せ名，トリガ識別子，文ラベル，ループ変数名，ホスト識別子，リスト名，ロール名，SQL パラメタ名） | 1 | 30 | バイト |
| FIX 表の行長 | 1 | 30,000 | バイト |
| 手続き又は関数の SQL パラメタ数 | 0 | 30,000 | 個 |
| 繰返し列数 | 2 | 30,000 | 個 |
| 1 サーバで生成するインデクス情報ファイル数 | ― | ―※1 | ― |
| 次に示すユティリティがメッセージに表示できる処理行数<br>• pdload<br>• pdrorg<br>• pdrbal | 0 | 4,294,967,295※2 | 件 |

注※1

プラグイン使用の有無，サーバ内の RD エリア数などによって最大値は変動します。なお，プラグインインデクスの遅延一括作成に HiRDB ファイルシステム領域を使用する場合，最大値は 4,096 になります。

注※2

4,294,967,295 件以上のデータを処理した場合，表示行数が一度 0 にリセットされ，再度 1 からカウントアップします。

## 付録 A.3　HiRDB ファイル名に関する最大値と最小値

HiRDB ファイル名に関する最大値と最小値を次の表に示します。

表 A-3　HiRDB ファイル名に関する最大値と最小値（単位：文字）

| HiRDB ファイル種別 | HiRDB ファイルに指定できるファイル名の長さ | | パス名の長さ※の最大値 |
|---|---|---|---|
| | 最小値 | 最大値 | |
| システムステータスファイル | 1 | 30 | 167 |
| サーバステータスファイル | 1 | 30 | 167 |
| システムログファイル | 1 | 30 | 167 |
| シンクポイントダンプファイル | 1 | 30 | 167 |
| アンロードログファイル | 1 | 30 | 167 |
| 監査証跡ファイル | 16 | — | 167 |
| 作業表用ファイル | 25 | — | 167 |
| RD エリア構成ファイル | 1 | 30 | 167 |
| バックアップファイル | 1 | 30 | 167 |
| アンロードデータファイル | 1 | 30 | 167 |
| インデクス情報ファイル | 30 | — | 167 |
| 差分バックアップ管理ファイル | 1 | 30 | 167 |

（凡例）—：該当しません。最小値の欄に記載されている値が固定値です。

注※

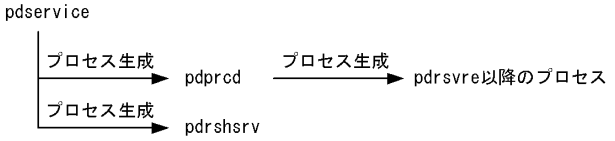
パス名の長さは,「HiRDB ファイルシステム領域 ¥HiRDB ファイル」の長さです。

# 付録 B　HiRDB のプロセス一覧

ここでは，HiRDB で起動するプロセスについて説明します。

## 付録 B.1　HiRDB/ シングルサーバで起動するプロセス

HiRDB で起動するプロセスの構造を次に示します。




HiRDB/ シングルサーバで起動するプロセスを次の表に示します。

**表 B-1　HiRDB/ シングルサーバで起動するプロセス**

| サーバ種別 | プロセス名 | | 説明 | プロセス数 | サーバ名称 |
|---|---|---|---|---|---|
| | マニュアルでの表記 | プロセス名称 | | | |
| システムサーバ | HiRDB サービスプロセス | pdservice | プロセスサーバの制御 | 1 | ― |
| | リモート受け付けプロセス | pdrshsrv | リモートシェル・リモートコピーの実行受け付け | 1（pdntenv コマンドでリモート機能を有効にした場合に起動します） | ― |
| | プロセスサーバプロセス | pdprcd | HiRDB 関連プロセスの管理 | 1 | _prc |
| | 後処理プロセス | pdrsvre | プロセス異常終了時の後始末処理 | pd_process_terminator の指定値が fixed の場合：pd_process_terminator_max の値<br>pd_process_terminator の指定値が resident の場合：1<br>pd_process_terminator の指定値が nonresident の場合：ダウンしたプロセスに対して 1 | _admrsvr |
| | サーバモード系切り替え用 HiRDB 起動プロセス | pdstart2d | クラスタソフトウェアと連動した HiRDB プロセスの起動制御<br>• スタンバイレス型系切り替え<br>• ユーザサーバホットスタンバイ<br>• 高速系切り替え | 1（オンライン時はなし） | _pdstrt2 |
| | メッセージログサーバプロセス | pdmlgd | メッセージ出力制御 | 1 | _mlg |

| サーバ種別 | プロセス名 | | 説明 | プロセス数 | サーバ名称 |
|---|---|---|---|---|---|
| | マニュアルでの表記 | プロセス名称 | | | |
| | システムマネジャプロセス | pdrdmd | ユニット起動・停止制御，接続ユーザ管理（なお，ネームサービス，又はノードマネジャと表記している場合があります） | 1 | _rdm |
| | ステータスサーバプロセス | pdstsd | ユニット用ステータスファイルの入出力制御 | 1 | _sts0 |
| | スケジューラプロセス | pdscdd | シングルサーバプロセスへのトランザクションの割り当て（なお，ロックサーバと表記している場合があります） | 1 | _scd |
| | トランザクションサーバプロセス | pdtrnd | トランザクション制御 | 1 | _trn |
| | トランザクション回復プロセス | pdtrnrvd | トランザクションの決着・回復制御 | 1 ～ダウンした pdsds 数※1 | _trnrcv |
| | 監査証跡管理サーバプロセス | pdaudd | 監査証跡管理 | pd_aud_file_name の指定がある場合は 1，ない場合は 0 | _aud |
| | 監査証跡自動データロード制御プロセス | pdaudld | 自動データロード用 pdload の起動制御 | pd_aud_file_name の指定があり，かつ pd_aud_auto_loading に Y を指定している場合は 1，それ以外は 0 | _audld |
| | ログサーバプロセス | pdlogd | システムログ取得制御，ログ関連プロセス制御 | 1 | _logN |
| | デファードライトプロセス | pd_buf_dfw | DB 格納ディスクへのバックグラウンドライト | 1 | ldfwN |
| | 非同期 READ プロセス | pd_ios_ard | 非同期 READ 機能 | pd_max_ard_process の値 | lardN |
| | デファードライト処理用並列 WRITE プロセス | pd_buf_awt | デファードライト処理の並列 WRITE 機能 | pd_dfw_awt_process の値 | lawtN |

| サーバ種別 | プロセス名 | | 説明 | プロセス数 | サーバ名称 |
|---|---|---|---|---|---|
| | マニュアルでの表記 | プロセス名称 | | | |
| | REDO プロセス | pd_rcv_rd | 全面リラン時の DB の前進復帰 | MIN（接続ディスク数，pd_max_recover_process の値）<br>接続ディスク数：RD エリアを定義した論理ボリューム数<br>pd_rdarea_open_attribute_use に Y を指定している場合は<br>pd_max_recover_process の値※2 | 2rrnM |
| | ログスワッパプロセス | pdlogswd | システムログ関連ファイルの割り当て・解放・入出力管理，シンクポイントダンプ取得 | 1 | _logNs |
| | デッドロック監視プロセス | pdlckmnd | 排他制御処理の分散を行っている場合のデッドロック検知 | pd_lck_pool_partition に 2 以上を指定し，かつ<br>pd_lck_deadlock_check に Y を指定すると 1，それ以外は 0 | _lckmnN |
| ユーザサーバ | シングルサーバプロセス | pdsds | SQL 処理 | pd_max_users の値※3 | サーバ名※4 |
| ユティリティサーバ※5 | pdinit 制御プロセス | pdinitd | 初期設定ユティリティ実行プロセス | 1 | 0minit0 |
| | pdcopy バックアップ出力プロセス | pdcopyb | バックアップファイル出力 | 多重度数×コマンド同時実行数 | 0bcpy?※6 |
| | pdcopy データベース読み込みプロセス | pdcopyr | データベース読み込み | コマンド同時実行数 | 0rcopy0 |
| | pdrstr バックアップ読み込みプロセス | pdrstrb | バックアップファイル読み込み | コマンド同時実行数 | 0brstr?0 |
| | pdrstr マスタディレクトリ用 RD エリア読み込みプロセス | pdrstrm | マスタディレクトリ用 RD エリア読み込み | コマンド同時実行数 | 0mrstr0 |
| | pdrstr アンロードログ読み込みプロセス | pdrstrl | アンロードログファイル読み込み | コマンド同時実行数 | 0lrstr0 |

| サーバ種別 | プロセス名 | | 説明 | プロセス数 | サーバ名称 |
|---|---|---|---|---|---|
| | マニュアルでの表記 | プロセス名称 | | | |
| | pdrstr データベース書き込みプロセス | pdrstrr | データベース書き込み | コマンド同時実行数 | 0rrstr0 |
| | pdrstr マスタディレクトリ用 RD エリア書き込みプロセス | pdrstrw | マスタディレクトリ用 RD エリア書き込み | コマンド同時実行数 | 0wrstr0 |
| | pdload 制御プロセス | pdloadm | データロード制御 | pdload コマンド同時実行数 | 0mload0 |
| | pdrorg 制御プロセス | pdrorgm | DB 再編成制御（アンロード，リロード，インデクス再編成・再作成，空きページ解放，及びグローバルバッファ常駐化） | pdrorg，pdreclaim，pdpgbfon コマンド同時実行数 | 0mrorg0 |
| | pdgetcst 制御プロセス | pdgcstm | 最適化情報収集 | pdgetcst コマンド同時実行数 | 0mgcst0 |
| | pddbst 制御プロセス | pddbstl | DB 状態解析制御 | pddbst コマンド同時実行数 | 0mdbst0 |
| | pdexp 制御プロセス | pdexpm | ディクショナリ搬出入制御 | 1 | 0mexp0 |
| | pdplgexe 制御プロセス | pdplgexe | プラグインユティリティ実行制御 | プラグインが提供するコマンド同時実行数 | 0mplge0 |
| | MIB パフォーマンス情報取得プロセス | pdmbcd | pdmibcmd コマンドによって指定された MIB パフォーマンス情報を取得する | 1 | 0mbcd0 |

（凡例）

－：該当しません。

注

- サーバ名称の xxxN はユーザサーバ数により 1，2，…ユニット最大サーバ数と増加します。
- サーバ名称の xxxM は定義により 2 ～ 11 と増加します。
- サーバ名称はメッセージ出力やコマンド情報出力などで使用されます。

注※ 1

ダウンした pdsds 数が二つ以上発生した場合に，ダウンした数と同数まで増加します。ダウンした pdsds のトランザクションが決着するに従ってプロセス数は減少し，回復対象となるトランザクションがなくなると 1 に戻ります。なお，上限値は pd_max_users の値× 2 ＋ 7 となります。

注※ 2

REDO プロセスは，HiRDB 開始時に起動され，起動が完了すれば停止します。

注※ 3

ユティリティサーバの場合にはプロセス数は 0 となります。計算後に 2000 を超える場合には 2000 までが起動されるプロセスとなります。また，pd_process_count が指定されている場合には起動プロセス数は指定値分となります。pd_process_count を超えるアクセス要求があった場合は，同時処理するために同時アクセス数のプロセスを pd_max_users の値まで起動します。

注※ 4

システム共通定義の pdstart オペランドの -s オプションで指定するサーバ名です。

注※ 5

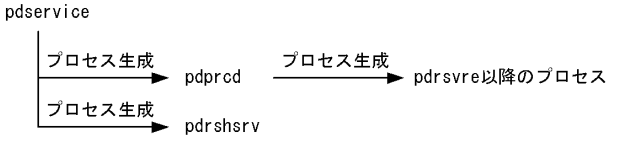
対応するコマンドが実行中のときだけプロセスが起動され，コマンドが終了すればプロセスは停止します。

注※ 6

「?」は，バックアップ出力プロセスの多重度数（pdcopy の -f オプションで指定した制御文ファイル中の -b オプション指定値数）により，0，1，…f と増加します。

## 付録 B.2　HiRDB/ パラレルサーバで起動するプロセス

HiRDB で起動するプロセスの構造を次に示します。




HiRDB/ パラレルサーバで起動するプロセスを表 B-2 及び表 B-3 に示します。

**表 B-2　HiRDB/ パラレルサーバで起動するプロセス（1/2）**

| サーバ種別 | プロセス名 | | 説明 | プロセス数 | サーバ名称 | サーバごとのプロセスの起動有無 | |
|---|---|---|---|---|---|---|---|
| | マニュアルでの表記 | プロセス名称 | | | | MGR | 非 MGR |
| システムサーバ | HiRDB サービスプロセス | pdservice | プロセスサーバの制御 | 1 | ― | ― | ― |
| | リモート受付プロセス | pdrshsrv | リモートシェル・リモートコピーの実行受付 | 1（pdntenv コマンドでリモート機能を有効にした場合に起動します） | ― | ― | ― |

| サーバ種別 | プロセス名 | | 説明 | プロセス数 | サーバ名称 | サーバごとのプロセスの起動有無 | |
|---|---|---|---|---|---|---|---|
| | マニュアルでの表記 | プロセス名称 | | | | MGR | 非 MGR |
| | プロセスサーバプロセス | pdprcd | HiRDB 関連プロセスの管理 | 1 | _prc | ○ | ○ |
| | 後処理プロセス | pdrsvre | プロセス異常終了時の後始末処理 | pd_process_terminator の指定値が fixed の場合：pd_process_terminator_max の値<br>pd_process_terminator の指定値が resident の場合：1<br>pd_process_terminator の指定値が nonresident の場合：ダウンしたプロセスに対して 1 | _admrsvr | ○ | ○ |
| | サーバモード系切り替え用 HiRDB 起動プロセス 1 | pdstart2d | クラスタソフトウェアと連動した HiRDB プロセスの起動制御<br>• スタンバイレス型系切り替え<br>• ユーザサーバホットスタンバイ<br>• 高速系切り替え<br>• 1：1 スタンバイレス型系切り替えの実行系 | 1（オンライン時はなし） | _pdstrt2 | ○ | ○ |
| | サーバモード系切り替え用 HiRDB 起動プロセス 2 | pdstart2a | クラスタソフトウェアと連動した HiRDB プロセスの起動制御<br>• 1：1 スタンバイレス型系切り替えの待機系 | 1（オンライン時はなし） | _pdst2a1 | × | ○ |
| | サーバモード系切り替え用 HiRDB 起動プロセス 3 | pdsvstartd | クラスタソフトウェアと連動した HiRDB プロセスの起動制御<br>• 影響分散スタンバイレス型系切り替えの待機系 | pd_ha_agent に activeunits を指定している場合は 1，それ以外は 0 | _pdsvstd | × | ○ |
| | メッセージログサーバプロセス | pdmlgd | メッセージ出力制御（pd_mlg_msg_log_unit に local を指定している場合に起動します） | 1 | _mlg | ○ | ○ |
| | システムマネジャ（MGR ユニット）プロセス | pdrdmd | ユニット起動・停止制御，接続ユーザ管理（なお，ネームサービス，又はノードマネジャと表記している場合があります） | 1 | _rdm | ○ | × |

| サーバ種別 | プロセス名 | | 説明 | プロセス数 | サーバ名称 | サーバごとのプロセスの起動有無 | |
|---|---|---|---|---|---|---|---|
| | マニュアルでの表記 | プロセス名称 | | | | MGR | 非 MGR |
| | ノードマネジャ（非 MGR ユニット）プロセス | pdndmd | ユニット起動・停止制御，接続ユーザ管理（なお，ネームサービスと表記している場合があります） | 1 | _ndm | × | ○ |
| | ステータスサーバプロセス | pdstsd | ユニット用ステータスファイルの入出力制御 | 1 | _sts0 | ○ | ○ |
| | スケジューラプロセス | pdscdd | バックエンドサーバプロセス，ディクショナリサーバプロセス，及びフロントエンドサーバプロセスの割り当て（なお，ロックサーバと表記している場合があります） | 1 | _scd | ○ | ○ |
| | トランザクションサーバプロセス | pdtrnd | トランザクション制御 | 1 | _trn | ○ | ○ |
| | トランザクション回復プロセス | pdtrnrvd | トランザクションの決着・回復制御 | FES の場合：<br>1 ～ダウンした pdfes 数※ 1<br>DS の場合：<br>1 ～ダウンした pddic 数※ 1<br>BES の場合：<br>1 ～ダウンした pdbes 数※ 1 | _trnrcv | ○ | ○ |
| | 監査証跡管理サーバプロセス | pdaudd | 監査証跡管理 | pd_aud_file_name の指定がある場合は 1，ない場合は 0 | _aud<br>_auz ※ 2 | ○ | ○ |
| | 監査証跡自動データロード制御プロセス | pdaudld | 自動データロード用 pdload の起動制御 | 監査証跡管理サーバプロセスの条件に合わせて pd_aud_auto_loading に Y を指定している場合は 1，N 又は指定していない場合は 0 | _audld | ○ | × |
| | ユニット監視プロセス | pdrdma | HiRDB ユニットが稼働しているかどうかの監視をおこなう | 1（ユニット数が 1 の場合は 0） | _rdmck | ○ | × |

表 B-3　HiRDB/ パラレルサーバで起動するプロセス（2/2）

| サーバ種別 | プロセス名 | | 説明 | プロセス数 | サーバ名称 | サーバごとのプロセスの起動有無 | | |
|---|---|---|---|---|---|---|---|---|
| | マニュアルでの表記 | プロセス名称 | | | | BES | DS | FES |
| システムサーバ | ログサーバプロセス | pdlogd | システムログ取得制御，ログ関連プロセス制御 | 1 | _logN<br>_lozN※2 | ○ | ○ | ○ |
| | デファードライトプロセス | pd_buf_dfw | DB 格納ディスクへのバックグラウンドライト | 1 | 1dfwN | ○ | ○ | × |
| | 非同期 READ プロセス | pd_ios_ard | 非同期 READ 機能 | pd_max_ard_process の値 | 1ardN | ○ | ○ | × |
| | 並列処理用デファード WRITE プロセス | pd_buf_awt | デファードライトの並列 WRITE 機能 | pd_dfw_awt_process の値 | 1awtN | ○ | ○ | × |
| | REDO プロセス | pd_rcv_rd | 全面リラン時の DB のロールフォワード | MIN（接続ディスク数，pd_max_recover_process の値）<br>接続ディスク数：RD エリアを定義した論理ボリューム数<br>pd_rdarea_open_attribute_use の値が Y の場合は<br>pd_max_recover_process の値※3 | 2rrnM | ○ | ○ | × |
| | ログスワッパプロセス | pdlogswd | システムログ関連ファイルの割り当て・解放・入出力管理，シンクポイントダンプ取得 | 1 | _logsN<br>_lozsN※2 | ○ | ○ | ○ |
| | デッドロック監視プロセス | pdlckmnd | 排他制御処理の分散を行っている場合のデッドロック検知 | pd_lck_pool_partition に 2 以上を指定し，かつ pd_lck_deadlock_check に Y を指定した場合は 1，それ以外は 0 | _lckmnN<br>_lckmzN※2 | ○ | ○ | ○ |
| ユーザサーバ | バックエンドサーバプロセス | pdbes | データベースへのアクセス | MAX（pd_max_users の値，pd_max_bes_process の値） | サーバ名※4 | ○ | × | × |
| | ディクショナリサーバプロセス | pddic | ディクショナリ表の一括管理 | MAX（pd_max_users の値，pd_max_dic_process の値） | サーバ名※4 | × | ○ | × |

| サーバ種別 | プロセス名 | | 説明 | プロセス数 | サーバ名称 | サーバごとのプロセスの起動有無 | | |
|---|---|---|---|---|---|---|---|---|
| | マニュアルでの表記 | プロセス名称 | | | | BES | DS | FES |
| | フロントエンドサーバプロセス | pdfes | SQL 処理，バックエンドサーバへの指示 | pd_max_users の値 | サーバ名※4 | × | × | ○ |
| ユティリティサーバ | pdinit 制御プロセス | pdinitd | 初期設定ユティリティ実行プロセス | 1 | 0minit0 | × | ○ | × |
| | pdinit 実行プロセス | pdinitb | 初期設定ユティリティ BES 側実行プロセス | 1 ～ 2 | 0sinit0 | ○ | × | × |
| | pdcopy バックアップ出力プロセス | pdcopyb | バックアップファイル出力（pdcopy に指定したバックアップ出力先（-b オプション）に起動します） | 多重度数× pdcopy コマンド同時実行数 | 0bcpy??※5 | × | × | × |
| | pdcopy データベース読み込みプロセス | pdcopyr | データベース読み込み | 複写対象サーバ数※6 | 0rcopyN※7 | ○ | ○ | × |
| | pdrstr バックアップ読み込みプロセス | pdrstrb | バックアップファイル読み込み（pdrstr に指定したバックアップ読み込み先（-b オプション）に起動します） | pdrstr コマンド同時実行数 | 0brstr?L※5※7 | × | × | × |
| | pdrstr マスタディレクトリ用 RD エリア読み込みプロセス | pdrstrm | マスタディレクトリ用 RD エリア読み込み※8 | pdrstr コマンド同時実行数 | 0mrstr | × | ○ | × |

| サーバ種別 | プロセス名 | | 説明 | プロセス数 | サーバ名称 | サーバごとのプロセスの起動有無 | | |
|---|---|---|---|---|---|---|---|---|
| | マニュアルでの表記 | プロセス名称 | | | | BES | DS | FES |
| | pdrstr アンロードログ読み込みプロセス | pdrstrl | アンロードログファイル読み込み※9 | pdrstr コマンド同時実行数 | 0lrstr | × | × | × |
| | pdrstr データベース書き込みプロセス | pdrstrr | データベース書き込み | 回復対象サーバ数※10 | 0rrstrL※7 | ○ | ○ | × |
| | pdrstr マスタディレクトリ用 RD エリア書き込みプロセス | pdrstrw | マスタディレクトリ用 RD エリア書き込み※11 | 1 | 0wrstr | × | ○ | × |
| | pdload 制御プロセス | pdloadm | データロード制御 | pdload コマンド同時実行数 | 0mload?※12 | ○ | ○ | ○ |
| | pdrorg 制御プロセス | pdrorgm | DB 再編成制御（アンロード，リロード，インデクス再編成・再作成，空きページ解放，及びグローバルバッファ常駐化） | pdrorg,pdreclaim,pdpgbfon コマンド同時実行数 | 0mrorg?※13 | ○ | ○ | ○ |
| | pdrbal 制御プロセス | pdrbalm | リバランス制御 | pdrbal コマンド同時実行数 | 0mrbal0 | × | ○ | × |
| | pdgetcst 制御プロセス | pdgcstm | 最適化情報収集 | pdgetcst コマンド同時実行数 | 0mgcst0 | × | ○ | × |
| | pddbst 制御プロセス | pddbstl | DB 状態解析制御（MGR ユニットで起動します） | pddbst コマンド同時実行数 | 0mdbst0 | × | × | × |
| | pdexp 制御プロセス | pdexpm | ディクショナリ搬出入制御（搬出ファイルがあるユニットに起動します） | 1 | 0mexp0 | × | × | × |

| サーバ種別 | プロセス名 | | 説明 | プロセス数 | サーバ名称 | サーバごとのプロセスの起動有無 | | |
|---|---|---|---|---|---|---|---|---|
| | マニュアルでの表記 | プロセス名称 | | | | BES | DS | FES |
| | pdplgexe 制御プロセス | pdplgexe | プラグインユティリティ実行制御 | プラグインが提供するコマンド同時実行数 | 0mplge0 | × | ○ | × |
| | MIB パフォーマンス情報取得プロセス | pdmbcd | pdmibcmd コマンドによって指定された MIB パフォーマンス情報を取得する（MGR ユニットで起動します） | 1 | 0mbcd0 | × | × | × |

（凡例）
○：プロセスが起動します。
×：プロセスが起動しません。
－：該当しません。

注
- サーバ名称の xxxN はユーザサーバ数により 1，2，…ユニット最大サーバ数と増加します。
- サーバ名称の xxxM は定義により 2 ～（ユニット最大サーバ数× 11）と増加します。

注※ 1
ダウンした pdbes，pdfes，pddic 数が二つ以上発生した場合に，ダウンした数と同数まで増加します。ダウンした pdbes，pdfes，pddic のトランザクションが決着するに従ってプロセス数は減少し，回復対象となるトランザクションがなくなると 1 に戻ります。上限値を次に示します。
FES の場合：pd_max_users の値× 2 ＋ 7
DS の場合：pd_max_dic_process の値× 2 ＋ 7）
BES の場合：pd_max_bes_process の値× 2 ＋ 7）

注※ 2
1：1 スタンバイレス型系切り替え構成の場合で，代替 BES ユニット用に起動するときの名称です。

注※ 3
REDO プロセスは，HiRDB 開始時に起動され，起動が完了すれば停止します。

注※ 4
システム共通定義の pdstart オペランドの -s オプションで指定するサーバ名です。

注※ 5
「?」は，バックアップ出力プロセスの多重度数（pdcopy の -f オプションで指定した制御文ファイル中の -b オプション指定値数）によって，0，1，…f と増加します。

注※ 6
pdcopy に指定した複写対象（-r，-s，-u，-a オプション）の RD エリアが属するサーバ数分起動します。

注※ 7

「N」は，コマンドごとに 1 から n の値が設定されます。

注※ 8

pdrstr に指定したバックアップファイル（-b オプション）があるホストと，ディクショナリサーバがあるホストが異なる場合に起動します。また，pdrstr に指定したバックアップファイル（-b オプション）の出力先ホストと，ディクショナリサーバがあるホストが異なる場合に起動します。

注※ 9

pdrstr でアンロードログファイル（-l オプション）又はディレクトリ（-d オプション）を指定した場合に，回復対象が 2 サーバ以上あるとき，若しくはアンロードログファイルが格納されているホストと回復対象の RD エリアが属するサーバがあるホストとが異なるときに起動します。

注※ 10

pdrstr に指定した回復対象（-r，-s，-u，-c，-a オプション）の RD エリアが属するサーバ数分起動します。

注※ 11

回復対象にマスタディレクトリ用 RD エリアを指定し，バックアップファイルがあるホストとディクショナリサーバがあるホストが異なる場合に起動します。

注※ 12

「?」は，pdload の source 制御文に指定したサーバがあるユニットによって，0，1，…f と増加します。

注※ 13

「?」は，pdrorg の場合は unload 制御文に指定したサーバがあるユニット，pdreclaim，pdpgbfon の場合は -s オプションに指定したサーバがあるユニットによって，0，1，…f と増加します。

# 付録Ｃ　Ｑ＆Ａ

HiRDB システムの構築に関する事例を Q&A 形式でまとめています。

次に示す項目について記載しています。

| 記載内容 | 参照箇所 |
|---|---|
| インストールディレクトリに関する質問 | (1) |
| 1073 エラーに関する質問 | (2) |
| 仮想メモリの見積もり方法に関する質問 | (3) |
| ネットワークドライブの使用に関する質問 | (4) |
| システム共通定義の pdstart オペランドに指定するホスト名に関する質問 | (5) |
| HiRDB/Developer's Kit に関する質問 | (6) |
| データベース定義ユティリティ（pddef）の実行に関する質問 | (7) |
| データベースの最大容量に関する質問 | (8) |
| OpenTP1 との XA インタフェースに関する質問 | (9) |
| FIX 表の性能に関する質問 | (10) |
| 重複キーインデクスに関する質問 | (11) |
| インデクスに関する質問 | (12) |
| 横分割表のインデクス定義に関する質問 | (13) |
| シンクポイントダンプの運用に関する質問 | (14) |
| ステータスファイルの運用に関する質問 | (15) ～ (18) |
| 作業表用 HiRDB ファイルシステム領域の最大使用量に関する質問 | (19) |
| pdstart コマンドで HiRDB が開始しない場合 | (20) |
| pdstart コマンドで特定のユニットが開始しない場合 | (21) |
| pdstart コマンドで HiRDB の開始が遅い場合 | (22) |
| pdstart コマンドがエラーリターンする場合（reason code= TIMEOUT） | (23) |
| データベース定義ユティリティ（pddef）がエラーとなる場合 | (24) |
| CREATE TABLE 文の LOB 列定義に関する質問 | (25) |
| ウィルス対策ソフトに関する質問 | (26) |

## (1) インストールディレクトリに関する質問

**質問**

インストールディレクトリについて，何か注意することはありますか？

**お答えします**

インストールディレクトリの名称を指定するときには，次に示す項目に注意してください。なお，インストールディレクトリのパス名は，環境変数 PDDIR に設定してあります。

- ドライブで始まり，次の要素で構成される文字列で指定してください。
  英数字
  _（下線）
  .（ピリオド）

パス区切りの ¥

- ドライブだけの指定はできません。
- 全角文字，及び特殊記号は使用しないでください。
- 長さは 128 バイト以内にしてください。
- システム共通定義の pdunit オペランドで指定する HiRDB 運用ディレクトリ名と，同じ名称にしてください。

### (2) 1073 エラーに関する質問

**質問**

インストール時に 1073 エラーになります。

**お答えします**

この 1073 エラーは，HiRDB のアンインストール後に OS を再起動しないで，再度 HiRDB をインストールしたときに発生します。HiRDB をインストール，又はアンインストールした後には，OS を再起動する必要があります。対処方法については，「2.2.5　インストール時に 1073 エラーになったときの対処」を参照してください。

### (3) 仮想メモリの見積もり方法に関する質問

**質問**

仮想メモリの見積もり方法について教えてください。

**お答えします**

一般的には実メモリの 2，3 倍程度ですが，仮想メモリはディスクを使用するため，大き過ぎるとディスクを圧迫することになります。これらを考慮して，見積もってください。
なお，実メモリを使用するか，又は仮想メモリを使用するかの制御は，OS が管理しています。HiRDB では制御できません。

### (4) ネットワークドライブの使用に関する質問

**質問**

ネットワークドライブは使用できますか？

**お答えします**

使用できません。HiRDB のインストールもそうですが，ユティリティのバックアップファイルなどもすべてローカルドライブを使用してください。

### (5) システム共通定義の pdstart オペランドに指定するホスト名に関する質問

**質問**

システム共通定義の pdstart オペランドに指定するホスト名は，どこの設定に対応しているのですか？

**お答えします**

［コントロールパネル］－［ネットワーク］の「ネットワークの設定」の TCP/IP のプロパティをクリックしてください。「DNS 設定」のホストに設定されている名称がホスト名です。この設定値は，コマンドプロンプトから hostname コマンドを実行しても確認できます。

### (6) HiRDB/Developer's Kit に関する質問

**質問**

HiRDB/Developer's Kit はどのようなときに必要ですか？

**お答えします**

HiRDB サーバがあるマシンで UAP を作成する場合は，HiRDB サーバに HiRDB/Developer's Kit の機能が含まれているので必要ありません。HiRDB サーバがあるマシンとは別のマシンで UAP を開発する場合に必要です。
また，HiRDB サーバと異なるプラットフォームの UAP を作成する場合にも必要です。

## （7）データベース定義ユティリティ（pddef）の実行に関する質問

**質問**

データベース定義ユティリティ（pddef）で CREATE TABLE を実行したのですが，何も実行されませんでした。なぜですか？

**お答えします**

pddef の制御文でセミコロン（;）の後ろに空白が入っていませんか？空白が入っていると，その SQL 文は実行されません（この例では，CREATE TABLE 以降の指定は省略しています）。
（誤）CREATE TABLE ; △　　△：空白
（正）CREATE TABLE ;

## （8）データベースの最大容量に関する質問

**質問**

HiRDB のデータベースの最大容量はどのくらいですか？

**お答えします**

一つの表は最大 512 個の RD エリアに分割格納でき，一つの RD エリアは最大 16 個の HiRDB ファイルで構成できます。

## （9）OpenTP1 との XA インタフェースに関する質問

**質問**

OpenTP1 と HiRDB の連携時，参照するだけの SQL であれば，そのトランザクションのコミットは XA インタフェースを通らないように思えますが，どうでしょうか？

**お答えします**

参照するだけの SQL であっても，コミット時には必ず XA インタフェースを経由して HiRDB に処理が渡されます。ただし，更新 SQL があるときのコミットに比べると，走行するステップ数が少なくなります。

## （10）FIX 表の性能に関する質問

**質問**

FIX 表とそうでない表（非 FIX 表）の性能差はどれくらいありますか？

**お答えします**

操作対象列数や操作対象行数によって性能差は変わるため一概には言えませんが，1 列の大量更新で FIX 指定をした場合の実行時間が，FIX 指定をしなかった場合に比べ，約 2/3 になった事例があります。
FIX 指定の方が性能劣化しないため，次に示す条件を満たす場合は FIX 指定にしてください。

- 可変長の列がない
- NULL 値を持つ列がない
- 列を追加しない

## (11) 重複キーインデクスに関する質問

**質問**

重複したキーに対してもインデクスを定義できますか？
定義できるなら，それによって何か困ることが生じるのでしょうか？

**お答えします**

インデクスは定義できます（非 UNIQUE 属性）。ただし，大量の重複キー（201 件以上）があるインデクスは特殊な格納構造になってアクセスするインデクスページが増えるため，性能上好ましくありません。

## (12) インデクスの使用方法に関する質問

**質問**

DBPARTNER2 から HiRDB をアクセスする形態で，ある表にインデクスを定義したとき，そのことを DBPARTNER2 側で意識する必要がありますか？

**お答えします**

ありません。DBPARTNER2 であっても通常のクライアント UAP であっても HiRDB から見ると同じ扱いになります。

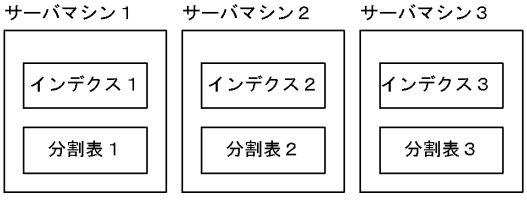
## (13) 横分割表のインデクス定義に関する質問

**質問**

サーバマシン間に分割された表に対してインデクスを定義する場合，どのようにインデクスを配置すればよいですか？

**お答えします**

次のように分割表単位にインデクスも分割してください。




## (14) シンクポイントダンプの運用に関する質問

**質問**

シンクポイントダンプファイルの有効保証世代数とはどのようなことですか？

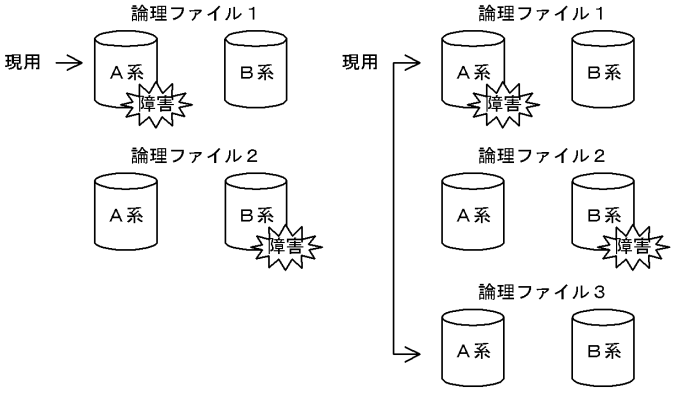
**お答えします**

シンクポイントダンプファイルには，全面回復処理に備えてシステムログファイルをどの位置から読み始めればよいかなどの情報をシンクポイントごとに取得します。シンクポイントダンプファイルに格納されている位置情報が示すシステムログファイル以降のシステムログファイルは，全面回復処理で使用する可能性があるため，上書き禁止状態にして保護しています。
有効保証世代数とは，「シンクポイントダンプファイルの何世代分の位置からシステムログファイルを上書き禁止にして保護するか」ということです。つまり，有効保証世代数が 1 の場合は，最新のシンクポイントダンプファイルが示すシステムログファイル以降が上書き禁止状態になります。有効保証世代数が 2 の場合は，最新のシンクポイントダンプファイルの 1 世代前のシンクポイントダンプファイルが示すシステムログファイル以降が上書き禁止状態になります。

## （15）ステータスファイルの運用に関する質問（ステータスファイルの二重化）

**質問**

ステータスファイルの二重化のペアはどのように構成するのですか？
次のように論理ファイル 1 の A 系と論理ファイル 2 の B 系が障害の場合，論理ファイル 2 の A 系と論理ファイル 1 の B 系でペアを組むことはありますか？




**お答えします**

異なる論理ファイル間でペアを組むことはありません。A 系，B 系が両方とも正常な論理ファイルにスワップします。A 系，B 系の両方とも正常な論理ファイルが一つもない場合は，pd_sts_singleoperation オペランドの指定に従ってユニットを異常終了するか，又は片系運転で処理を続行します。

## （16）ステータスファイルの運用に関する質問（障害発生時）

**質問**

A 系，B 系の両方とも正常な論理ファイルが一つもない（A 系，B 系のどちらかが障害）場合の処理方式を決める次に示すオペランドがあります。

- pd_syssts_singleoperation = stop | continue（ユニット用ステータスファイルの場合）
- pd_sts_singleoperation = stop | continue（サーバ用ステータスファイルの場合）

stop と continue，どちらを指定すればよいですか？

**お答えします**

stop を指定すると，HiRDB（HiRDB/ パラレルサーバの場合はユニット）が異常終了します。
continue を指定すると，ステータスファイルを片系運転します。
ステータスファイルは，全面回復処理のための情報を記録している重要なファイルです。continue を指定して片系運転中にステータスファイルが障害になると，両系障害でユニットが異常終了します。そして，現用ファイルが両系ともアクセスできないため，全面回復処理ができなくなります。したがって，次のような考え方で判断してください。

- HiRDB の異常終了よりも，全面回復処理の保証を重視する場合は stop を指定します。
- とにかく HiRDB を途中で停止したくない（最悪の場合，全面回復処理はあきらめ，データベースをバックアップ時点まで戻したりデータロードし直す）場合は continue を指定します。

## （17）ステータスファイルの運用に関する質問（ステータスファイルの定義）

**質問**

論理ステータスファイルは1～7個定義できますが，ディスクに余裕がありません。どの程度用意するのが現実的ですか？

**お答えします**

ディスクを障害回復して再使用可能になるまでの安全性を考えると，論理ステータスファイルは三つ（二重化× 3=6 ファイル）以上用意することをお勧めします。

ディスクに余裕がなければ，論理ステータスファイルは二つ（二重化× 2=4 物理ファイル）が妥当です。この場合，障害発生時には速やかにステータスファイルを回復してください。

論理ステータスファイルを一つしか用意しないと，ステータスファイルの障害発生時にデータベースをバックアップから回復する必要があります。

## (18) ステータスファイルの運用に関する質問（ステータスファイルの配置）

**質問**

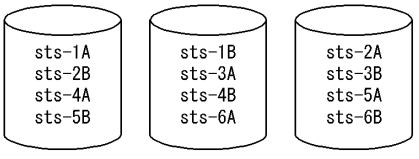
ステータスファイルはどのようにディスクに配置すればよいですか？

**お答えします**

基本は，複数の物理ステータスファイルを同じディスクに配置しないことです。同じディスクに配置すると，二重化や論理ファイルの複数化の意味がなくなります。つまり，論理ファイルを二つ定義している場合は四つのディスクに分散配置し，論理ファイルを三つ定義している場合は六つのディスクに分散配置します。

なお，次のように，物理ステータスファイルをリング状に配置すると，少ないディスク数で信頼性を確保できます。

**論理ファイル数 6 の場合の配置例**




このように配置すると，1ボリュームに障害が発生しても，2世代の両系ファイルが無事のままです。

## (19) 作業表用 HiRDB ファイルシステム領域の最大使用量に関する質問

**質問**

作業表用ファイルを作成する HiRDB ファイルシステム領域（バックエンドサーバ定義の pdwork オペランド）の最大使用量を知りたいのですが，手段がありますか？

**お答えします**

pdfstatfs コマンドで取得できます。

pdfstatfs -d **作業表用** HiRDB **ファイルシステム領域名**

「-d」は，HiRDB ファイルシステム領域に割り当てた領域の最大使用値を表示するオプションです。出力の「peak capacity」が最大使用値です。

なお，上記の最大使用値は，pdfstatfs コマンドでクリアできます。

pdfstatfs -c **作業表用** HiRDB **ファイルシステム領域名**

「-c」は，HiRDB ファイルシステム領域に割り当てた領域の最大使用値を 0 にするオプションです。

**注意事項**

pdfstatfs コマンドの「-d，-c」オプションは，HiRDB ファイルシステム領域の使用目的が「WORK」

又は「UTL」の場合に有効です。使用目的が「SYS，DB」の場合は表示されません。
なお，HiRDB ファイルシステム領域の使用目的は，pdfmkfs コマンドの -k オプションで指定します。

## （20）pdstart コマンドで HiRDB が開始しない場合

**質問**

pdstart コマンドで HiRDB が開始しないで，Psp4017 で -prc がアボートしてしまうのはなぜですか？

**お答えします**

次に示す原因が考えられます。

- HiRDB が正しくセットアップされていない

HiRDB を正しくセットアップし直してください。

## （21）pdstart コマンドで特定のユニットが開始しない場合

**質問**

pdstart コマンドを実行しましたが，システムマネジャ以外のユニットが開始しないのはなぜですか？

**お答えします**

開始できなかったユニットのシステム共通定義を確認してください。pdunit 又は pdstart オペランドの指定値が，システムマネジャがあるユニットのシステム共通定義と一致していない可能性があります。
開始できなかったユニットのシステム共通定義を修正した後に，pdstart -u コマンドでユニットを再開始してください。

## （22）pdstart コマンドで HiRDB の開始が遅い場合

**質問**

pdstart コマンドを入力して，コマンドが KFPS05078-I Unable to recognize HiRDB initialization Completion で終了しましたが，すべてのユニットの開始に時間が掛かるのはなぜですか？
1 時間～ 2 時間ぐらいで開始しました。

**お答えします**

1. KFPS00608-W（-314）メッセージが複数出力されている場合は，すべてのユニットのシステム共通定義を確認してください。pdunit 及び pdstart オペランドで指定したホスト名がすべて一致しているか，正しいホスト（存在するホスト）名が指定されているかを確認してください。
2. HiRDB で指定したホスト，ネットワークがすべて開始完了（稼働中）になっているかを確認してください。

## （23）pdstart コマンドがエラーリターンする場合（reason code= TIMEOUT）

**質問**

pdstart コマンドが KFPS01861-E メッセージ（reason code=TIMEOUT）を出力してエラーリターンしてしまいます。なぜですか？

**お答えします**

次に示す原因が考えられます。

1. ユニット開始に予想以上に時間が掛かってしまった
2. サーバ共通定義又は各サーバ定義の指定に誤りがある

次に示す対策をしてください。

1. pd_start_time_out オペランドの指定値を大きくしてから，pdstart コマンドを再入力してください。
2. イベントログに出力されている HiRDB のメッセージを参照して，誤っている定義を修正してください。

## (24) データベース定義ユティリティ（pddef）がエラーになる場合

**質問**

データベース初期設定ユティリティ（pdinit）は実行できましたが，データベース定義ユティリティ（pddef）がエラーになって実行できません。どんな原因が考えられますか？

**お答えします**

次に示す原因が考えられます。

- **応答なしや接続エラーになる場合は，環境変数の設定漏れが考えられます。**
  PDHOST，PDNAMEPORT の設定値を確認してください。

PDHOST：

HiRDB サーバがあるホスト名を指定します。システム共通定義の pdstart オペランドに指定したホスト名を指定します。

PDNAMEPORT：

システム共通定義の pd_name_port オペランドに指定したポート番号を指定します。

- **コネクトエラーとなる場合，環境変数 PDUSER の設定値の不正が考えられます。**
  pdinit の実行直後は，DBA 権限を持つ認可識別子が一つだけ存在します。認可識別子及びパスワードを hirdb.ini ファイルの環境変数 PDUSER に指定してください。

**注意事項**

1. pdinit 実行直後の認可識別子とパスワードについては，マニュアル「HiRDB Version 8 コマンドリファレンス」の「データベース初期設定ユティリティ（pdinit）」の「オプション」を参照してください。
2. 環境変数 PDUSER を設定するときは，認可識別子とパスワードを引用符で囲み，その外側をアポストロフィで囲んでください。これは，ほかの HiRDB のユティリティを実行する場合や，UAP を実行する場合も同じです。

## (25) CREATE TABLE 文の LOB 列定義に関する質問

**質問**

CREATE TABLE 文の列定義で，LOB 列の最大長を指定した場合（例：300 メガバイト）と指定しなかった場合（省略値 2 ギガバイト）では，HiRDB サーバと HiRDB クライアントのメモリ所要量やデータ転送量にどのような違いがありますか？

**お答えします**

LOB 列の最大長を指定しても指定しなくても，メモリ所要量やデータ転送量は同じです。LOB 列の検索又は更新時に使用するメモリ所要量やデータ転送量は，列定義の最大長ではなく，検索又は更新時の実長と埋込み変数の定義長に依存します。なお，格納するバイナリデータサイズを制限したい場合は，LOB 列の最大長で制限できます。

## (26) ウィルス対策ソフトに関する質問

**質問**

ウィルス対策ソフトをインストールしたところ，UAP が HiRDB に接続できなくなりました。ウィル

ス対策ソフトのファイアウォールが原因のようです。ファイアウォールの除外リストに，どのポート番号を除外指定すればよいでしょうか？

**お答えします**

「23.5.1　HiRDB で指定するポート番号の一覧」に HiRDB で使用するポート番号の一覧があります。この一覧の中で，次に示すポート番号を除外指定してください。

なお，ファイアウォールを設置した場所によって，除外指定するポート番号が異なります。

**● HiRDB サーバ側にファイアウォールを設置した場合**

システム共通定義及びユニット制御情報定義で指定する次に示すポート番号を除外指定してください。

- HiRDB のポート番号
- スケジューラプロセスのポート番号

これらのポート番号を指定するオペランドを指定していない場合は，オペランドを指定して，除外指定してください。

**●クライアント側にファイアウォールを設置した場合**

クライアント環境定義で指定する次に示すポート番号を除外指定してください。

- クライアントの受信ポート番号

このポート番号を指定するクライアント環境定義を指定していない場合は，クライアント環境定義を指定して，除外指定してください。

# 付録 D　バッチファイルによる環境設定

## 付録 D.1　バッチファイルによる環境設定の概要

ここでは，バッチファイルによる環境設定の方法について説明します。

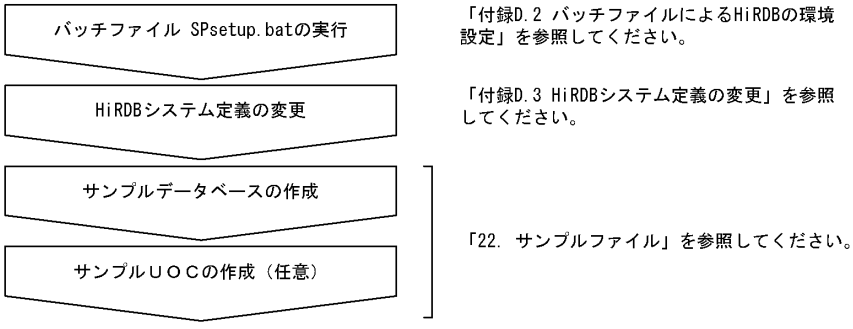
**この章を読むときの注意**

バッチファイルによる環境設定は，HiRDB/ シングルサーバのときだけ使えます。そのため，この章での説明は「HiRDB/ **シングルサーバをデフォルトのインストールディレクトリ（C:¥win32app¥hitachi¥hirdb_s）にインストールした**」と仮定して説明しています。

HiRDB/ パラレルサーバの環境設定については，%PDDIR%¥HiRDEF¥readme.txt を参照してください。

バッチファイルによる環境設定の手順を次の図に示します。

**図 D-1　バッチファイルによる環境設定の手順**




### (1) バッチファイル（SPsetup.bat）の内容

HiRDB では，HiRDB の環境設定を自動的に実行するバッチファイル SPsetup.bat（C:¥win32app¥hitachi¥hirdb_s¥sample¥SPsetup.bat）を提供しています。SPsetup.bat を実行すると，次に示す作業を自動的に実行します。

- HiRDB システム定義ファイルの作成
- HiRDB ファイルシステム領域の作成
- システムファイルの作成
- RD エリアの作成

この SPsetup.bat は次の表に示すファイルを使用して環境を自動生成します。これらのファイルを**サンプルコンフィグレーション**といいます。次の表に示すファイルの内容をカスタマイズすれば，業務に合わせたシステムを構築できます。これらのファイルは，C:¥win32app¥hitachi¥hirdb_s¥conf ディレクトリ下にあります。

**表 D-1　サンプルコンフィグレーションの内容**

| ファイル名 | 説　明 |
|---|---|
| pdsys<br>pdutsys<br>sds01 | HiRDB システム定義ファイル<br>• pdsys：システム共通定義<br>• pdutsys：ユニット制御情報定義<br>• sds01：シングルサーバ定義 |

| ファイル名 | 説　明 |
|---|---|
| fmkfile.bat | 次に示す HiRDB ファイルシステム領域を作成するバッチファイル<br>• RD エリア用の HiRDB ファイルシステム領域<br>• 作業表用ファイル用の HiRDB ファイルシステム領域 |
| fmkfs.bat | 次に示す HiRDB ファイルシステム領域を作成するバッチファイル<br>• システムファイル用の HiRDB ファイルシステム領域 |
| sysfint.bat | システムファイルを作成するバッチファイル |
| initdb.bat | RD エリアを作成するバッチファイルです。データベース初期設定ユティリティ（pdinit）を実行するバッチファイル |
| mkinit | データベース初期設定ユティリティ（pdinit）の制御文を格納しているバッチファイル |

## (2) 構築されるシステムの構成

SPsetup.bat を実行して作成されるシステムの構成例を図 D-2 及び図 D-3 に示します。ここで示すのは，バッチファイルをカスタマイズしない場合に作成されるシステム構成です。

**図 D-2　SPsetup.bat を実行して作成されるシステムの構成例**

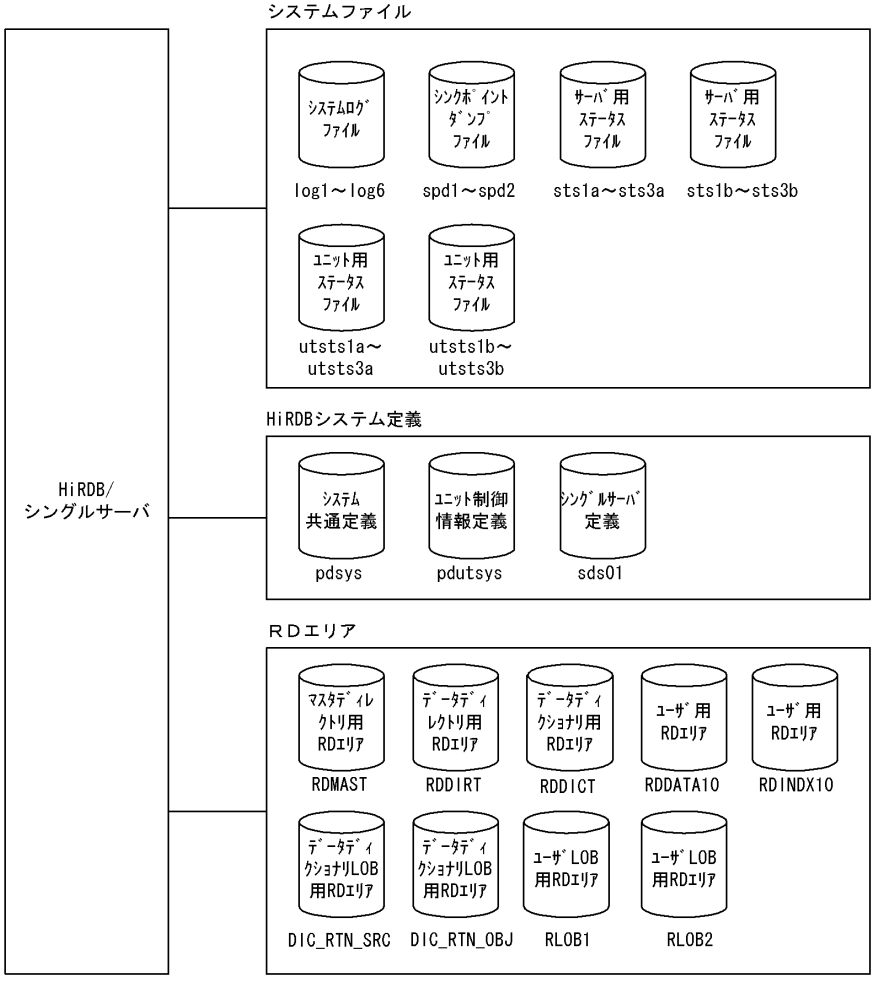
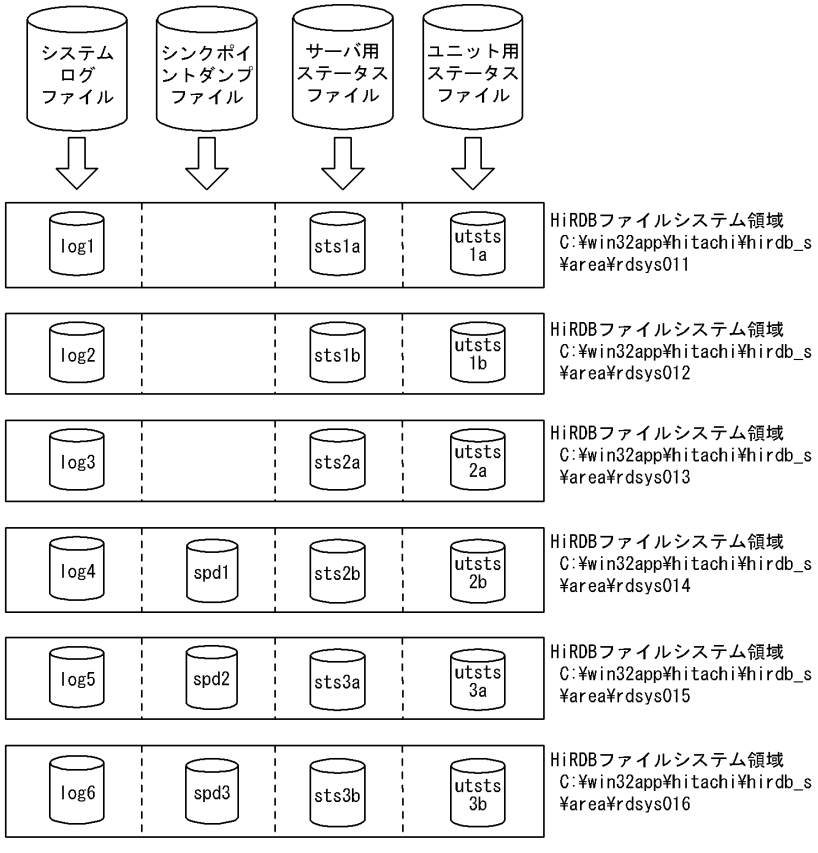
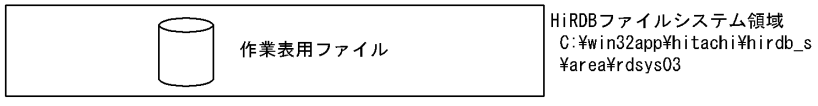


**図 D-3　SPsetup.bat を実行して作成される HiRDB ファイルシステム領域の構成例**

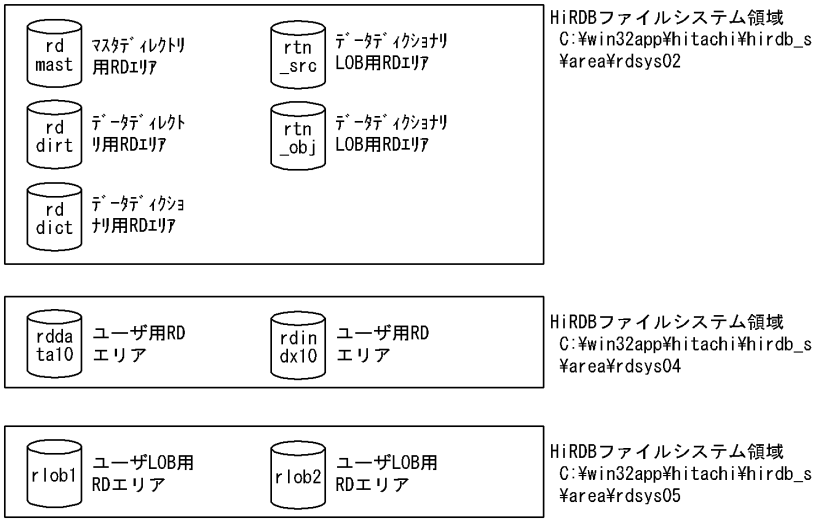
● システムファイル用のHiRDBファイルシステム領域




● 作業表用ファイル用のHiRDBファイルシステム領域



● ＲＤエリア用のHiRDBファイルシステム領域




## (3) 作成される HiRDB ファイルシステム領域及びファイル

SPsetup.bat を実行して作成される HiRDB ファイルシステム領域とファイルを表 D-2 及び表 D-3 に示します。ここで示すのは，バッチファイルをカスタマイズしない場合に作成される HiRDB ファイルシステム領域とファイルです。

**表 D-2　SPsetup.bat を実行して作成される HiRDB ファイルシステム領域名とサイズ**

| HiRDB ファイルシステム領域の種類 | HiRDB ファイルシステム領域のサイズ（単位：メガバイト） | HiRDB ファイルシステム領域名 |
|---|---|---|
| システムファイル用 | 小規模→ 74<br>中規模→ 148<br>大規模→ 296 | rdsys011 ※ 1<br>rdsys012 ※ 1<br>rdsys013 ※ 1<br>rdsys014 ※ 1<br>rdsys015 ※ 1<br>rdsys016 ※ 1 |
| RD エリア用（システム用 RD エリア） | 小規模→ 20<br>中規模→ 40<br>大規模→ 80 | rdsys02 ※ 2 |
| 作業表用ファイル用 | － | rdsys03 ※ 2 |
| RD エリア用（ユーザ用 RD エリア） | 小規模→ 40<br>中規模→ 80<br>大規模→ 160 | rdsys04 ※ 2 |
| RD エリア用（ユーザ LOB 用 RD エリア） | 40 | rdsys05 ※ 2 |

注※ 1

SPsetup.bat 実行時に表示されるセットアップ内容の確認画面の［HiRDB システムファイル領域用ディレクトリ］で指定したディレクトリ下に作成されます。省略値は %PDDIR%¥area になります。

注※ 2

SPsetup.bat 実行時に表示されるセットアップ内容の確認画面の［HiRDB RD エリア領域用ディレクトリ］で指定したディレクトリ下に作成されます。省略値は %PDDIR%¥area ディレクトリになります。

**表 D-3　SPsetup.bat を実行して作成されるファイルの名称**

| ファイルの種類 | | ファイル名 | 備考 |
|---|---|---|---|
| HiRDB システム定義ファイル | システム共通定義ファイル | %PDDIR%¥conf¥pdsys | ディレクトリはインストール時に作成されます。 |
| | ユニット制御情報定義ファイル | %PDDIR%¥conf¥pdutsys | |
| | シングルサーバ定義ファイル | %PDDIR%¥conf¥sds01 | |
| システムファイル | システムログファイル | rdsys011¥log1 ※1<br>rdsys012¥log2 ※1<br>rdsys013¥log3 ※1<br>rdsys014¥log4 ※1<br>rdsys015¥log5 ※1<br>rdsys016¥log6 ※1 | 6 グループ |
| | シンクポイントダンプファイル | rdsys014¥spd1 ※1<br>rdsys015¥spd2 ※1<br>rdsys016¥spd3 ※1 | 3 グループ |
| | ユニット用ステータスファイル | rdsys011¥utsts1a ※1<br>rdsys012¥utsts1b ※1<br>rdsys013¥utsts2a ※1<br>rdsys014¥utsts2b ※1<br>rdsys015¥utsts3a ※1<br>rdsys016¥utsts3b ※1 | ユニットごとに<br>二重化× 3 個 |
| | サーバ用ステータスファイル | rdsys011¥sts1a ※1<br>rdsys012¥sts1b ※1<br>rdsys013¥sts2a ※1<br>rdsys014¥sts2b ※1<br>rdsys015¥sts3a ※1<br>rdsys016¥sts3b ※1 | サーバごとに<br>二重化× 3 個 |
| システム用 RD エリア | マスタディレクトリ用 RD エリア | rdsys02¥rdmast ※2 | RD エリア名は RDMAST |
| | データディレクトリ用 RD エリア | rdsys02¥rddirt ※2 | RD エリア名は RDDIRT |
| | データディクショナリ用 RD エリア | rdsys02¥rddict ※2 | RD エリア名は RDDICT |
| | データディクショナリ LOB 用 RD エリア（ソース格納用） | rdsys02¥rtn_src ※2 | RD エリア名は DIC_RTN_SRC |
| | データディクショナリ LOB 用 RD エリア（オブジェクト格納用） | rdsys02¥rtn_obj ※2 | RD エリア名は DIC_RTN_OBJ |
| 作業表用ファイル | | rdsys03 ※2 | ― |
| RPC トレースファイル | | %PDDIR%¥spool¥pdrpctr | ― |

| ファイルの種類 | ファイル名 | 備考 |
|---|---|---|
| ユーザ用 RD エリア（データ格納用） | rdsys04¥rddata10 ※2 | RD エリア名は RDDATA10 |
| ユーザ用 RD エリア（インデクス格納用） | rdsys04¥rdindx10 ※2 | RD エリア名は RDINDX10 |
| ユーザ LOB 用 RD エリア | rdsys05¥rlob1 ※2 | RD エリア名は RLOB1 |
| | rdsys05¥rlob2 ※2 | RD エリア名は RLOB2 |

注※ 1

SPsetup.bat 実行時に表示されるセットアップ内容の確認画面の［HiRDB システムファイル領域用ディレクトリ］で指定したディレクトリ下に作成されます。省略値は %PDDIR%¥area ディレクトリになります。

注※ 2

SPsetup.bat 実行時に表示されるセットアップ内容の確認画面の［HiRDB RD エリア領域用ディレクトリ］で指定したディレクトリ下に作成されます。省略値は %PDDIR%¥area ディレクトリになります。

## 付録 D.2　バッチファイルによる HiRDB の環境設定

ここでは，HiRDB/**シングルサーバの環境を設定**する例で説明します。

**実行者**　HiRDB **管理者**

（1）～（8）の手順に従って HiRDB の環境設定をします。

### （1）SPsetup.bat を実行します

C:¥win32app¥hitachi¥hirdb_s¥sample¥SPsetup.bat をエクスプローラで表示して，アイコンをダブルクリックしてください。

「**既存の定義ファイルはありません**」という［HiRDB システム定義］メッセージボックスが表示されます（実行するのが 2 回目以降の場合は表示されません）。［HiRDB システム定義］メッセージボックスの【OK】ボタンをクリックしてください。

### （2）HiRDB の規模を選択します

［HiRDB システム定義］ダイアログボックスが表示されます。［ファイル］－［新規セットアップ］を選択してください。［HiRDB 環境定義］の簡易セットアップ画面が表示されます。

ディスク容量と仮想メモリの値を確認してから，実行する規模を選択します。選択したら［OK］ボタンをクリックしてください。

**HiRDB の規模を選択するときの注意**

大規模，中規模，小規模を選択すると，それぞれ pd_max_users（最大同時接続ユーザ数）に 32，16，8 が設定されたシステム共通定義が作成されます。購入した HiRDB の最大同時接続ユーザライセンスの範囲で選択してください。

### (3) セットアップの内容を確認します

〔説明〕

**システムファイルのフィールド**

- ［HiRDB システムファイル領域用ディレクトリ］には，システムファイルを作成するディレクトリ名を指定します。
- 簡易セットアップでは，システムログファイルの二重化は指定できません。そのため，［ログファイルを 2 重化する］チェックボックスはグレー表示になります。

**RD エリアのフィールド**

- ［HiRDB RD エリア領域用ディレクトリ］には，RD エリアを作成するディレクトリ名を指定します。

### (4) セットアップを開始します

［セットアップ内容の確認］画面の内容を確認したら，**［セットアップ開始］**をクリックしてください。コマンドプロンプトに次に示す内容が表示されます。

```
"＃＃＃　hirdb.iniを更新しますか。"
"？？？　（開始＝Enter）or（終了＝N）"
"＝＝＝＝＞"
```

Enter **キーを押してください。**Enter キーを押すと，メモ帳で hosts ファイルの内容が表示されます。hosts ファイルに，このコンピュータの IP アドレスとホスト名があるかどうかを確認してください。ない場合は，このコンピュータの IP アドレスとホスト名を追加してください。

確認したら（又は追加したら），メモ帳を終了してください。終了すると，hirdb.ini ファイルが作成されます。

IP **アドレスが分からない場合**

**［コントロールパネル］**の**［ネットワーク］**をダブルクリックして，［ネットワーク］の**［プロトコル］**タブを選択します。［TCP/IP プロトコル］のプロパティを開き，その中の［IP アドレス］で確認してください。

### (5) HiRDB を初期化します

HiRDB を初期化します。コマンドプロンプトに次に示す内容が表示されます。

```
"＃＃＃　HiRDBシステムの初期化を行います"
"＃＃＃　（インストールディレクトリ名）下に必要なファイルがある場合は，移動してください"
"？？？　（開始＝Enter）or（終了＝N）"
"＝＝＝＝＞"
```

Enter **キーを押してください。**コマンドプロンプトに次に示す内容が表示されます。

```
"＃＃＃　終了します。"
"続行するときは何かキーを押してください"
```

**何かキーを押してください。**コマンドプロンプトが順に 3 画面表示され，次に示す内容が表示されます。

```
"続行するときは何かキーを押してください"
```

**何かキーを押してください。**これで，次に示すファイルが作成されます。

- HiRDB システム定義ファイル
- システムファイル

HiRDB システム定義ファイルは，C:¥win32app¥hitachi¥hirdb_s¥conf 下に次に示すファイル名で作成されます。

- システム共通定義：pdsys
- ユニット制御情報定義：pdutsys
- シングルサーバ定義：sds01

### (6) HiRDB を開始します

［HiRDB システム定義］メッセージボックスに［HiRDB **を開始しますか？**］というメッセージが表示されたら，［OK］ボタンをクリックしてください。クリックすると，コマンドプロンプトに次に示す内容が表示されて，HiRDB が開始されます。

```
"HiRDB/Single Serverサービスを開始します"
"HiRDB/Single Serverサービスは正常に開始されました。"
```

### (7) データベース初期設定ユティリティ（pdinit）が実行されます

データベース初期設定ユティリティ（pdinit）が実行されて RD エリアが作成されます。データベース初期設定ユティリティ（pdinit）が終了すると，コマンドプロンプトに KFPX24000-I メッセージ（リターンコードが 0）が表示され，さらに次に示す内容が表示されます。

```
"続行するときは何かキーを押してください"
```

**何かキーを押してください。**ここまでの操作が終了すると，［HiRDB システム定義］メッセージボックスに「**セットアップ終了。アプリケーションを終了します。**」というメッセージが表示され，HiRDB のシステム定義が終了します。

**注意事項**

HiRDB のメッセージを確認して，問題がなければ Enter キーを押してください。問題がある場合は強制停止の運用コマンド（pdstop -f コマンド）で HiRDB を強制停止してください。その後で問題を解決して，再度実行してください。

### (8) HiRDB の稼働状況を確認します

実際に HiRDB が稼働しているかどうかを pdls コマンドで確認します。HiRDB の運用コマンドは，コマンドプロンプトから入力します。

```
C:¥>pdls

HOSTNAME(112852)                UNITID SVID     STATUS    STARTTIME
k95x620                         unt1   ******** ACTIVE    111130
k95x620                         unt1   sds01    ACTIVE    111130
```

〔説明〕

STATUS が ACTIVE になっていれば，HiRDB が稼働中です。

これで HiRDB サーバのシステム構築が終わりました。必要に応じて RD エリアを作成して，表を定義してください。

# 付録 D.3　HiRDB システム定義の変更

**実行者**　HiRDB 管理者

「付録 D.2　バッチファイルによる HiRDB の環境設定」で説明した作業を終えると，HiRDB システム定義が作成されています。必要に応じて，HiRDB システム定義を変更してください。HiRDB システム定義を変更する場合は，HiRDB を正常終了させてください。その後，HiRDB システム定義を変更してください。

HiRDB システム定義変更（作成時）の注意を次に示します。

**注意事項**

1. HiRDB システム定義ファイルのパーミッションは，ファイルの所有者（HiRDB 管理者）にだけ，読み込み権限及び書き込み権限を持たせるように設定，維持するようにしてください。
2. システム共通定義の pdunit オペランドを指定する場合，HiRDB 運用ディレクトリには環境変数 PDDIR と同じ指定をしてください。大文字及び小文字も同じ指定にしてください。
3. システム共通定義の pdstart オペランドと pdunit オペランドの -x に指定するホスト名は同じにしてください。そのホスト名は，コマンドプロンプトで hostname を実行すると得られます。
4. システム共通定義の TZ オペランド（タイムゾーン）の指定（省略時は JST-9）と，サーバマシンのシステム環境変数 TZ が同じであることを確認してください。コマンドプロンプトで set コマンドを入力すると，サーバマシンの環境変数が表示されます。システム環境変数 TZ が未設定又は内容がシステム共通定義の TZ オペランドと異なる場合は，HiRDB の動作が保障できません。システム環境変数は［コントロールパネル］の［システム］をダブルクリックして，［システムのプロパティ］の［環境］タブで切り替わる画面に設定します。
5. HiRDB システム定義ファイルには，.txt などの拡張子を付けないでください。

# 索引

## 数字

1073 エラーに関する質問 809
1073 エラーになったとき〔インストール〕 57
24 時間連続稼働 2

## A

Administrators 権限 54
AFTER トリガ 383
ALTER TRIGGER 383
ARRAY オプション〔CREATE TABLE〕 419

## B

B-tree 構造のインデクスの設計 463
BEFORE トリガ 383
BINARY 型 400
BLOB 型 400
BLOB 型データの検索又は更新時に必要なメモリ所要量〔HiRDB/ シングルサーバ〕 537
BLOB 型データの検索又は更新時に必要なメモリ所要量〔バックエンドサーバ又はディクショナリサーバ〕 594
BLOB 型データの検索又は更新時に必要なメモリ所要量〔フロントエンドサーバ〕 593

## C

CLUSTER KEY オプション〔CREATE TABLE〕 395
CREATE INDEX 463
create rdarea 文 165，170，503
CREATE SCHEMA 198
CREATE TRIGGER 383
CREATE TYPE 230

## D

DBPARTNER2 から HiRDB をアクセスする形態 811
DECIMAL 型の符号正規化機能 199
DROP TRIGGER 383
DTP 249

## E

EBCDIK 404
ESIS-B 形式 217
EUC 中国語漢字コード 58
EXCEPT VALUES オプション〔CREATE INDEX〕 469

## F

FES ホストダイレクト接続機能 297
FIX 属性の指定 393
FIX ハッシュ分割 362
FIX 表の性能に関する質問 810
FOREIGN KEY 431
FQDN を指定した HiRDB サーバへの接続方法 770

## H

HiRDB/Developer's Kit に関する質問 809
HiRDB/ シングルサーバのアンインストール 63
HiRDB/ シングルサーバのメモリ所要量の見積もり 506
HiRDB/ パラレルサーバのアンインストール 64
HiRDB/ パラレルサーバのメモリ所要量の見積もり 540
HiRDB Advanced Partitioning Option 378
HiRDB Dataextractor との連携 248
HiRDB Datareplicator との連携 248
HiRDB External Data Access 機能 4，266
HiRDB Text Search Plug-in 212
HiRDB XA ライブラリ 249
HiRDB XA ライブラリが提供する機能 250
HiRDB XML Extension 215
HiRDB 運用ディレクトリ下のファイルの削除 58
HiRDB が作成するディレクトリ及びファイル 6
HiRDB が使用するポート数 783
HiRDB 管理者が作成するディレクトリ及びファイル 6
HiRDB 管理者の登録〔インストール〕 51
HiRDB 管理者の登録〔マルチ HiRDB〕 324
HiRDB サーバと HiRDB クライアントの接続方法 770
HiRDB システム定義（UAP 環境定義を除く）の変更方法 144
HiRDB システム定義の作成 136
HiRDB システム定義の作成〔HiRDB/ シングルサーバ〕 136
HiRDB システム定義の作成〔HiRDB/ パラレルサーバ〕 138
HiRDB システム定義の設定〔マルチ HiRDB〕 325
HiRDB システム定義の変更 826

HiRDB システム定義ファイルの構成例〔HiRDB/ シングルサーバ〕 138
HiRDB システム定義ファイルの構成例〔HiRDB/ パラレルサーバ〕 142
HiRDB のインストール手順 54
HiRDB の開始が遅い場合 814
HiRDB の環境設定の概要 2
HiRDB のサービスの停止 30
HiRDB の状態確認〔バージョンアップ〕 30
HiRDB の初期開始 169
HiRDB のディレクトリ及びファイル構成 6
HiRDB のバージョンアップ 28
HiRDB のメモリ所要量 505
HiRDB ファイルシステム領域サイズの見積もり〔作業表用ファイル〕 697
HiRDB ファイルシステム領域のアクセス権の考え方 60
HiRDB ファイルシステム領域の最大長 147，277，306
HiRDB ファイルシステム領域の作成 147
HiRDB ファイルシステム領域の作成〔raw I/O 機能〕 152
HiRDB ファイルシステム領域の作成〔RD エリア用〕 149
HiRDB ファイルシステム領域の作成〔作業表用ファイル用〕 150
HiRDB ファイルシステム領域の作成〔システムファイル用〕 150
HiRDB ファイルシステム領域の作成〔ユティリティ用〕 151
HiRDB ファイルシステム領域の作成〔リスト用 RD エリア用〕 151
HiRDB ファイルシステム領域の種類 147
HiRDB ファイルシステム領域の設計〔HiRDB/ シングルサーバ〕 275
HiRDB ファイルシステム領域の設計〔HiRDB/ パラレルサーバ〕 303
HiRDB ファイルシステム領域の設計〔RD エリア用〕 275，303
HiRDB ファイルシステム領域の設計〔作業表用ファイル用〕 276，304
HiRDB ファイルシステム領域の設計〔システムファイル用〕 275，304
HiRDB ファイルシステム領域の設計〔ユティリティ用〕 276，305
HiRDB ファイルシステム領域の設計〔リスト用 RD エリア用〕 277，306
HiRDB ファイルシステム領域を作成する準備 60
HiRDB ファイル名に関する最大値と最小値 794
HiRDB 予約ポート機能 789
HiRDB をインストールするときの注意〔マルチ HiRDB〕 324
Hub 最適化情報定義の作成 142

## I

INSERT ONLY オプション〔CREATE TABLE〕 407
IP アドレス 770

## J

Java 仮想マシンが使用するメモリ所要量〔HiRDB/ シングルサーバ〕 510
Java 仮想マシンが使用するメモリ所要量〔HiRDB/ パラレルサーバ〕 545
JP1/AJS2 270
JP1/Automatic Job Management System 2 267
JP1/Base 267
JP1/Base に通知できる HiRDB のイベント 267
JP1/IM 269
JP1/IM によるイベント監視の概要 269
JP1/IM によるイベント管理 269
JP1/Integrated Management - Manager 267

## K

KFPS01861-E メッセージ 814
KFPS05078-I メッセージ 814

## L

LOB 用グローバルバッファの割り当て 332
LOB 用グローバルバッファを割り当てる場合 333
LOB 列 400
LOB 列を定義した表の作成 209
LRU 管理方式 341

## N

NAT が設置されている場合の設定 778
NO SPLIT オプション 399

## O

OLTP との連携 249
OLTP と連携した HiRDB システムの構成例 251
OLTP と連携できる製品 249
OpenTP1 249
OpenTP1 との XA インタフェースに関する質問 810
OS 環境ファイルの設定〔インストール〕 52
OTS 250

## P


PATH 56
PCTFREE オプション〔CREATE TABLE〕489
PCTFREE オプション〔CREATE TABLE 又は CREATE INDEX〕492
pd_assurance_table_no オペランド 499
pd_buf_awt〔HiRDB/ シングルサーバ〕797
pd_buf_awt〔HiRDB/ パラレルサーバ〕803
pd_buf_dfw〔HiRDB/ シングルサーバ〕797
pd_buf_dfw〔HiRDB/ パラレルサーバ〕803
pd_check_pending オペランド 430，453
pd_dbbuff_lru_option オペランド 341
pd_dbbuff_rate_updpage オペランド 339
pd_dbsync_point オペランド 338，340
pd_dfw_awt_process オペランド 339
pd_foreign_server_libpath 266
pd_hb_ary_fec_num オペランド 588
pd_ios_ard〔HiRDB/ シングルサーバ〕797
pd_ios_ard〔HiRDB/ パラレルサーバ〕803
pd_jp1_event_level オペランド 267
pd_jp1_use オペランド 267
pd_large_file_use 147，278，306
pd_log_dual オペランド 281，309
pd_log_rec_leng オペランド 281，309
pd_log_rerun_reserved_file_open オペランド 281，309
pd_log_rpl_no_standby_file_opr オペランド 248
pd_log_singleoperation オペランド 281，309
pd_max_foreign_server 266
pd_ntfs_cache_disable 278，306
pd_pageaccess_mode オペランド 346
pd_rcv_rd〔HiRDB/ シングルサーバ〕798
pd_rcv_rd〔HiRDB/ パラレルサーバ〕803
pd_rdarea_warning_point_msgout オペランド 500
pd_registered_port オペランド 783，789
pd_rpl_hdepath オペランド 248
pd_rpl_init_start オペランド 248
pd_shared_rdarea_use オペランド 502
pd_spd_assurance_count オペランド 283，311
pd_spd_dual 283，311
pd_spd_reduced_mode オペランド 284，312
pd_spd_reserved_file_auto_open オペランド 284，312
pd_spool_cleanup_interval 58
pd_spool_cleanup_interval_level 58
pd_spool_cleanup_level オペランド 58
pd_spool_cleanup オペランド 58
pd_sts_singleoperation オペランド 286，314
pd_sysdef_default_option オペランド 31
pd_syssts_singleoperation オペランド 286，314
pdadmvr コマンド 39
pdaudd〔HiRDB/ シングルサーバ〕797
pdaudd〔HiRDB/ パラレルサーバ〕802
pdaudld〔HiRDB/ シングルサーバ〕797
pdaudld〔HiRDB/ パラレルサーバ〕802
pdbes〔HiRDB/ パラレルサーバ〕803
pdbuffer オペランド 333
pdbufmod コマンド 330
pdchgconf コマンド 145
pdconfchk コマンド 136
PDCONFPATH 55
pdcopyb〔HiRDB/ シングルサーバ〕798
pdcopyb〔HiRDB/ パラレルサーバ〕804
pdcopyr〔HiRDB/ シングルサーバ〕798
pdcopyr〔HiRDB/ パラレルサーバ〕804
pdcspool コマンド 58
pddbstl〔HiRDB/ シングルサーバ〕799
pddbstl〔HiRDB/ パラレルサーバ〕805
pddef の制御文 810
pddic〔HiRDB/ パラレルサーバ〕803
PDDIR 55
pdexpm〔HiRDB/ シングルサーバ〕799
pdexpm〔HiRDB/ パラレルサーバ〕805
pdfes〔HiRDB/ パラレルサーバ〕804
pdfmkfs コマンド 147，502
pdgcstm〔HiRDB/ シングルサーバ〕799
pdgcstm〔HiRDB/ パラレルサーバ〕805
pdhubopt 266
pdinit 165
pdinitb〔HiRDB/ パラレルサーバ〕804
pdinitd〔HiRDB/ シングルサーバ〕798
pdinitd〔HiRDB/ パラレルサーバ〕804
pdlckmnd〔HiRDB/ シングルサーバ〕798
pdlckmnd〔HiRDB/ パラレルサーバ〕803
pdload 199
pdloadm〔HiRDB/ シングルサーバ〕799
pdloadm〔HiRDB/ パラレルサーバ〕805
pdlogd〔HiRDB/ シングルサーバ〕797
pdlogd〔HiRDB/ パラレルサーバ〕803
pdloginit コマンド 153
pdlogswd〔HiRDB/ シングルサーバ〕798
pdlogswd〔HiRDB/ パラレルサーバ〕803
pdls -d ust コマンド〔HiRDB のバージョンアップ時〕30
pdls コマンド 41
pdmbcd〔HiRDB/ シングルサーバ〕799
pdmbcd〔HiRDB/ パラレルサーバ〕806
pdmlgd〔HiRDB/ シングルサーバ〕796
pdmlgd〔HiRDB/ パラレルサーバ〕801

pdmod 170
pdndmd〔HiRDB/ パラレルサーバ〕802
pdntenv コマンド 51，52，58
pdpgbfon 348
pdplgexe〔HiRDB/ シングルサーバ〕799
pdplgexe〔HiRDB/ パラレルサーバ〕806
pdplgrgst コマンド 187
pdplugin オペランド 190
pdprcd〔HiRDB/ シングルサーバ〕796
pdprcd〔HiRDB/ パラレルサーバ〕801
pdprgcopy コマンド 41
pdprgrenew コマンド 41
pdrbalm〔HiRDB/ パラレルサーバ〕805
pdrdma〔HiRDB/ パラレルサーバ〕802
pdrdmd〔HiRDB/ シングルサーバ〕797
pdrdmd〔HiRDB/ パラレルサーバ〕801
pdrorgm〔HiRDB/ シングルサーバ〕799
pdrorgm〔HiRDB/ パラレルサーバ〕805
pdrplstart コマンド 299
pdrplstop コマンド 299
pdrshsrv〔HiRDB/ シングルサーバ〕796
pdrshsrv〔HiRDB/ パラレルサーバ〕800
pdrstrb〔HiRDB/ シングルサーバ〕798
pdrstrb〔HiRDB/ パラレルサーバ〕804
pdrstrl〔HiRDB/ シングルサーバ〕798
pdrstrl〔HiRDB/ パラレルサーバ〕805
pdrstrm〔HiRDB/ シングルサーバ〕798
pdrstrm〔HiRDB/ パラレルサーバ〕804
pdrstrr〔HiRDB/ シングルサーバ〕799
pdrstrr〔HiRDB/ パラレルサーバ〕805
pdrstrw〔HiRDB/ シングルサーバ〕799
pdrstrw〔HiRDB/ パラレルサーバ〕805
pdrsvre〔HiRDB/ シングルサーバ〕796
pdrsvre〔HiRDB/ パラレルサーバ〕801
pdscdd〔HiRDB/ シングルサーバ〕797
pdscdd〔HiRDB/ パラレルサーバ〕802
pdsds〔HiRDB/ シングルサーバ〕798
pdservice〔HiRDB/ シングルサーバ〕796
pdservice〔HiRDB/ パラレルサーバ〕800
pdstart2a〔HiRDB/ パラレルサーバ〕801
pdstart2d〔HiRDB/ シングルサーバ〕796
pdstart2d〔HiRDB/ パラレルサーバ〕801
pdstart オペランドに指定するホスト名 809
pdstart コマンドがエラーリターンする場合 814
pdstart コマンドで HiRDB が開始しない場合 814
pdstart コマンドで特定のユニットが開始しない場合 814
pdstbegin 266
pdstsd〔HiRDB/ シングルサーバ〕797
pdstsd〔HiRDB/ パラレルサーバ〕802
pdstsinit コマンド 154
pdsvstartd〔HiRDB/ パラレルサーバ〕801
pdtrnd〔HiRDB/ シングルサーバ〕797
pdtrnd〔HiRDB/ パラレルサーバ〕802
pdtrnrvd〔HiRDB/ シングルサーバ〕797
pdtrnrvd〔HiRDB/ パラレルサーバ〕802
PDUXPLDIR 56
PDUXPLMSGMNI 746，749
PDUXPLMSGTQL 746，749
PDUXPLSEMMAX 746，750
PDUXPLSHMMAX 746，750
PRIMARY KEY オプション〔CREATE TABLE〕394
PRIVATE 423
PROTECTED 423
Psp4017 814
PUBLIC 423

## Q

Q & A 808

## R

raw I/O 機能 147
raw I/O 機能を使用した HiRDB ファイルシステム領域の作成 147
raw I/O 機能を使用した HiRDB ファイルシステム領域の作成〔例題〕152
RD エリアに関する最大値・最小値 485
RD エリアの自動増分機能使用時に出力されるシステムログ量 682
RD エリアの設計 483
RD エリアの配置〔HiRDB/ シングルサーバ〕288
RD エリアの配置〔HiRDB/ パラレルサーバ〕316
RD エリアの容量の見積もり 597
RD エリア用の HiRDB ファイルシステム領域の作成 149
RD エリア用の HiRDB ファイルシステム領域の設計〔HiRDB/ シングルサーバ〕275
RD エリア用の HiRDB ファイルシステム領域の設計〔HiRDB/ パラレルサーバ〕303
RD エリアを設計するときの検討項目 484
reason code= TIMEOUT 814
RECOVERY オペランド〔CREATE TABLE〕236，200
RM 249
RM 関連オブジェクト名 259
RM スイッチ名 256
RM 名 257

## S

sampleDB1.bat 763
sampleDB2.bat 763
sampleDB3.bat 763
sampleDB4.bat 763
SEGMENT REUSE オプション 499
SEGMENT REUSE オプション〔ALTER TABLE〕 499
server name オペランド 503
SPsetup.bat 817
SQL 関連の注意事項〔X/Open XA インタフェース環境〕 264
SQL 実行時に必要なメモリ所要量〔HiRDB/ シングルサーバ〕 529
SQL 実行時に必要なメモリ所要量〔HiRDB/ パラレルサーバ〕 585
SQL 操作に応じて出力されるシステムログ量 682
SQL 文が使用する作業表用ファイルの容量 697
SQL 前処理時に必要なメモリ所要量〔HiRDB/ シングルサーバ〕 536
SQL 前処理時に必要なメモリ所要量〔HiRDB/ パラレルサーバ〕 591
SQL 予約語定義の作成〔HiRDB/ シングルサーバ〕 137
SQL 予約語定義の作成〔HiRDB/ パラレルサーバ〕 141
Sun Java System Directory Server 4
SUPPRESS オプション〔CREATE TABLE〕 397

## T

TIMEOUT 814
TM 249
TPBroker for C++ 249
trnstring オペランド 254
TUXEDO 249
TZ の確認〔インストール〕 48

## U

UAP 環境定義の作成〔HiRDB/ シングルサーバ〕 137
UAP 環境定義の作成〔HiRDB/ パラレルサーバ〕 140
UAP 環境定義の追加又は変更方法 146
Unicode 58
UOC 205
UTF-8 58
UTF16 404

## W

WebLogic Server 249
Windows ファイアウォールの設定を有効にしている場合 61
Windows ファイアウォールの例外リストからの削除 69
Windows ファイアウォールの例外リストへの登録 66
WITHOUT ROLLBACK オプションの指定 405

## X

X/Open XA インタフェース 249
xa_switch_t 構造体名 256
XA インタフェース〔マルチスレッド対応〕 250
XML 型 215
XML 型全文検索用インデクス（n-gram） 217

## あ

空きセグメント 487
空きページ再利用モード 496
空き領域の確認〔バージョンアップ〕 28
空き領域の再利用機能 496
空き領域の再利用機能のページサーチ空回り回数 500
アクセスパス表示ユティリティ（pdvwopt）実行時のファイルの容量 721
アクセスパス表示ユティリティ（pdvwopt）実行時のメモリ所要量 738
新しいユーザを追加する場合 764
アンインストール 63
アンインストール〔プラグイン〕 195

## い

一意性制約 394
位置情報作業表の最大数の求め方 701
位置情報作業表の容量の求め方 700
一相最適化 250
一相最適化に関する注意事項 264
イベント属性定義ファイル 269
イベント通知〔JP1 との連携〕 267
インストーラ〔修正版 HiRDB への入れ替え〕 38
インストール 47
インストール〔付加 PP〕 62
インストール〔プラグイン〕 186
インストール〔マルチ HiRDB〕 324
インストール後の作業 58
インストール後の注意 56
インストールディレクトリに関する質問 808
インストール手順 54
インストール前に必要な作業 48
インストール前の注意 54

インデクス 463
インデクス一括作成中に発生したエラーの対処 238
インデクス格納 RD エリアの容量見積もり〔注意事項〕 485
インデクス定義時に出力されるシステムログ量 669
インデクスの格納ページ数 609，641
インデクスの格納ページ数の計算例 613
インデクスのキー長一覧 612
インデクスのキー長の上限 463
インデクスの作成 463
インデクスの使用方法に関する質問 811
インデクスの設計 461
インデクスの定義 234
インデクスの分割指針 472
インデクスの優先順位 465
インデクスの横分割 471
インデクスページスプリット 677
インデクスページスプリット時のインデクスログ量の見積もり 677
インデクス用グローバルバッファの割り当て 330
インデクス用グローバルバッファを割り当てる場合 333
インデクス用ローカルバッファ 350
インデクス用ローカルバッファの割り当て 350
インデクスログ量の見積もり 676
インデクスを設計するときの検討項目 462
インデクスを定義できないデータ型 463
隠蔽レベル 231，423
インメモリデータ処理で必要なメモリ所要量〔HiRDB/ パラレルサーバ〕 539，595
インメモリデータバッファが使用する共用メモリセグメント数 539，596

## う

ウィザードセットアップ 90
ウィルス対策ソフト 815

## お

オープン文字列 257
オンライン状態であるかどうかの確認〔バージョンアップ〕 30

## か

改竄防止機能 407
改竄防止表 407
解析情報表及び運用履歴表を格納するデータディクショナリ用 RD エリアの容量の見積もり 649
外部キー 430
回復不要 FES 298
回復不要 FES ユニット 298
外部サーバ情報定義の作成 141
外部サーバ接続用のバックエンドサーバ 144
外部表検索時に必要なメモリ所要量 588
各サーバが使用する共用メモリの計算式〔HiRDB/ パラレルサーバ〕 572
拡張属性〔JP1 との連携〕 267
格納条件指定 361
カスタムセットアップ 95
カスタムセットアップ（詳細定義） 97
仮想メモリの確認〔インストール〕 50
仮想メモリの見積もり方法に関する質問 809
片系運転〔HiRDB/ シングルサーバ〕 281，286
片系運転〔HiRDB/ パラレルサーバ〕 309，314
簡易セットアップツールによる環境設定 71
環境設定〔簡易セットアップツール〕 72
環境変数〔HiRDB/ シングルサーバのリソース数〕 746
環境変数〔HiRDB/ パラレルサーバのリソース数〕 749
環境変数〔インストール〕 55
環境変数設定用バッチファイルの使用方法〔マルチ HiRDB〕 327
環境変数の設定〔マルチ HiRDB〕 325
監査証跡ファイルの容量の見積もり 693

## き

キーレンジ分割 361
キーレンジ分割（格納条件指定）の例 366
キーレンジ分割（境界値指定）の例 367
既定の照合順 404
既定文字集合 404
基本行ログ量の見積もり 672
基本属性〔JP1 との連携〕 267
旧値相関名 383
旧バージョンと新バージョンを入れ替える場合 32
旧バージョンに戻す場合 36
境界値指定 361
行削除禁止期間 408
共用 RD エリア 502
共用表 424
共用メモリ〔メモリ所要量〕 509，544
共用メモリ使用数 746，750

## く

空白変換機能 199
クライアントからの接続用ポート番号の設定〔マルチ HiRDB〕 327

クライアント環境定義〔トランザクションマネジャに登録〕 259
クライアント環境定義の設定〔データベースの作成〕 198
クライアント環境定義の設定〔マルチ HiRDB〕 326
クライアントとの接続 770
クラスタキーの指定 395
繰返し列 418
繰返し列のデータ長の求め方 606
繰返し列を含む表 418
グループ分け高速化機能実行時に必要なメモリ所要量〔HiRDB/ シングルサーバ〕 529
グループ分け高速化機能実行時に必要なメモリ所要量〔HiRDB/ パラレルサーバ〕 585
クローズ文字列 259
グローバルバッファが使用する共用メモリ〔HiRDB/ シングルサーバ〕 527
グローバルバッファが使用する共用メモリ〔HiRDB/ パラレルサーバ〕 579
グローバルバッファ常駐化ユティリティ 348
グローバルバッファの LRU 管理方式 341
グローバルバッファの先読み入力 348
グローバルバッファの設計 329
グローバルバッファの定義例 333
グローバルバッファの動的変更 330
グローバルバッファのバッファ面数の設定 335
グローバルバッファの割り当て 330
グローバルバッファの割り当て方法 333

## け

系切り替え機能 4
系切り替え機能との関連〔マルチ HiRDB〕 327
継承 231，420
検査制約 453
検査制約表 453
検査制約用解析ツリー長 639
検査保留状態 438，454

## こ

更新可能バックエンドサーバ 424
更新可能列 407
更新バッファ 330
更新前ログ取得モード 200，235
更新ログ取得方式の種類 199
高速接続機能 297
候補キー 394
コストベースの最適化 464
コマンドによる環境設定 133
コマンドによる環境設定の概要 134
コマンドを別名で実行するためのバッチファイルの作成 765
コミット時反映処理 340
コンストラクタ関数 231，422

## さ

サーバが使用する共用メモリの計算式〔HiRDB/ シングルサーバ〕 523
サーバが使用する共用メモリの計算式〔HiRDB/ パラレルサーバ〕 572
サーバ間横分割 475
サーバ共通定義の作成 139
サーバ内横分割 474
サーバマシン環境〔インストール〕 48
サービスポート番号の OS への登録〔マルチ HiRDB〕 324
最小値 792
最初に作成するファイル 6
最大使用量〔作業表用 HiRDB ファイルシステム領域〕 813
最大増分回数の見積もり〔作業表用ファイル〕 706
最大値 792
最大ファイル数の見積もり〔作業表用ファイル〕 704
最大容量に関する質問〔データベース〕 810
最適化情報収集ユティリティ（pdgetcst）実行時のファイルの容量 720
最適化情報収集ユティリティ（pdgetcst）実行時のメモリ所要量 733
作業表用 HiRDB ファイルシステム領域の最大使用量に関する質問 813
作業表用ファイル 696
作業表用ファイルの容量の見積もり 695
作業表用ファイル用の HiRDB ファイルシステム領域の作成 150
作業表用ファイル用の HiRDB ファイルシステム領域の設計〔HiRDB/ シングルサーバ〕 276
作業表用ファイル用の HiRDB ファイルシステム領域の設計〔HiRDB/ パラレルサーバ〕 304
作業用のコンソールの使用方法〔マルチ HiRDB〕 327
サブタイプ 420
サプレスオプションの指定 397
参照制約 430
参照専用バックエンドサーバ 424
参照バッファ 330
参照表 430
サンプル UOC のファイル名 760
サンプルオーディットのファイル名 758
サンプルコンフィグレーション 817

サンプルコンフィグレーションのファイル名 759
サンプルデータベースのカスタマイズ 763
サンプルデータベースの作成手順 763
サンプルデータベースのファイル名 759
サンプルファイル 757
サンプルファイルの使用方法 763

## し

システム環境変数 TZ の確認〔インストール〕 48
システムキャッシュの確認〔インストール〕 51
システム共通定義の作成〔HiRDB/ シングルサーバ〕 136
システム共通定義の作成〔HiRDB/ パラレルサーバ〕 138
システム構成〔HiRDB/ シングルサーバ〕 273
システム構成〔HiRDB/ パラレルサーバ〕 294
システム構成に関する最小値 792
システム構成に関する最大値 792
システム構成変更コマンド 145
システム構築手順 2
システム設計〔HiRDB/ シングルサーバ〕 272
システム設計〔HiRDB/ パラレルサーバ〕 292
システム設計〔マルチ HiRDB〕 324
システムの構成の確認〔マルチ HiRDB〕 324
システムファイルの作成 153
システムファイルの作成例〔HiRDB/ シングルサーバ〕 155
システムファイルの作成例〔HiRDB/ パラレルサーバ〕 157
システムファイルの設計〔HiRDB/ シングルサーバ〕 279
システムファイルの設計〔HiRDB/ パラレルサーバ〕 307
システムファイルの容量の見積もり 665
システムファイル用の HiRDB ファイルシステム領域の作成 150
システムファイル用の HiRDB ファイルシステム領域の設計〔HiRDB/ シングルサーバ〕 275
システムファイル用の HiRDB ファイルシステム領域の設計〔HiRDB/ パラレルサーバ〕 304
システムマネジャの設置 292
システム用 RD エリアの作成 165
システム用 RD エリアの配置〔HiRDB/ シングルサーバ〕 288
システム用 RD エリアの配置〔HiRDB/ パラレルサーバ〕 316
システム用 RD エリアのバックアップの取得〔バージョンアップ〕 29
システムログファイルの空き容量監視機能〔HiRDB/ シングルサーバ〕 279
システムログファイルの空き容量監視機能〔HiRDB/ パラレルサーバ〕 307
システムログファイルの片系運転〔HiRDB/ シングルサーバ〕 281
システムログファイルの片系運転〔HiRDB/ パラレルサーバ〕 309
システムログファイルの作成 153
システムログファイルの自動オープン〔HiRDB/ シングルサーバ〕 281
システムログファイルの自動オープン〔HiRDB/ パラレルサーバ〕 309
システムログファイルの設計〔HiRDB/ シングルサーバ〕 279
システムログファイルの設計〔HiRDB/ パラレルサーバ〕 307
システムログファイルの総容量 666
システムログファイルの総レコード数の確認〔バージョンアップ〕 31
システムログファイルの二重化〔HiRDB/ シングルサーバ〕 281
システムログファイルの二重化〔HiRDB/ パラレルサーバ〕 309
システムログファイルの容量の見積もり 666
システムログファイルの両系運転〔HiRDB/ シングルサーバ〕 281
システムログファイルの両系運転〔HiRDB/ パラレルサーバ〕 309
システムログファイルのレコード長〔HiRDB/ シングルサーバ〕 281
システムログファイルのレコード長〔HiRDB/ パラレルサーバ〕 309
システムログ量の求め方 666
実表 391
自動オープン〔HiRDB/ シングルサーバ〕 281，284
自動オープン〔HiRDB/ パラレルサーバ〕 309，312
自動採番機能を使用したデータロード 203
シフト JIS 漢字コード 58
修正パッチ〔修正版 HiRDB への入れ替え〕 38
修正版 HiRDB 38
修正版 HiRDB への入れ替え 38
主キー 394
縮退運転〔HiRDB/ シングルサーバ〕 283
縮退運転〔HiRDB/ パラレルサーバ〕 312
順序数生成子 203
障害時の運用〔修正版 HiRDB への入れ替え〕 44
詳細定義情報 72
使用中空きセグメント 487
使用中空きページ 490

使用中空きページの解放 493
使用中セグメント 487
使用中ページ 490
使用中満杯ページ 490
除外キー値 469
除外キー値を設定したインデクスの使用 469
新規ページ追加モード 496
新旧値別名 383
シンクポイントダンプの運用に関する質問 811
シンクポイントダンプファイルの作成 153
シンクポイントダンプファイルの自動オープン〔HiRDB/ シングルサーバ〕 284
シンクポイントダンプファイルの自動オープン〔HiRDB/ パラレルサーバ〕 312
シンクポイントダンプファイルの縮退運転〔HiRDB/ シングルサーバ〕 283
シンクポイントダンプファイルの縮退運転〔HiRDB/ パラレルサーバ〕 312
シンクポイントダンプファイルの設計〔HiRDB/ シングルサーバ〕 282
シンクポイントダンプファイルの設計〔HiRDB/ パラレルサーバ〕 310
シンクポイントダンプファイルの二重化 282，310
シンクポイントダンプファイルの有効保証世代数〔HiRDB/ シングルサーバ〕 283
シンクポイントダンプファイルの有効保証世代数〔HiRDB/ パラレルサーバ〕 311
シンクポイントダンプファイルの容量の見積もり 686
シンクポイントダンプファイルのレコード数の求め方 686
シングルサーバが使用する共用メモリ 523
シングルサーバ定義の作成 137
新値相関名 383

## す

スーパタイプ 420
スキーマの定義〔データベースの作成〕 198
ステータスファイルの運用に関する質問〔障害発生時〕 812
ステータスファイルの運用に関する質問〔ステータスファイルの定義〕 812
ステータスファイルの運用に関する質問〔ステータスファイルの二重化〕 812
ステータスファイルの運用に関する質問〔ステータスファイルの配置〕 813
ステータスファイルの片系運転〔HiRDB/ シングルサーバ〕 285
ステータスファイルの片系運転〔HiRDB/ パラレルサーバ〕 314
ステータスファイルの作成 154
ステータスファイルの設計〔HiRDB/ シングルサーバ〕 284
ステータスファイルの設計〔HiRDB/ パラレルサーバ〕 312
ステータスファイルの容量の見積もり 687
ステータスファイルの両系運転〔HiRDB/ シングルサーバ〕 286
ステータスファイルの両系運転〔HiRDB/ パラレルサーバ〕 314
ステータスファイルのレコード数の求め方 687
スナップショット方式 346

## せ

正規化 357
整合性チェックユティリティ（pdconstck）実行時のファイルの容量 723
静的登録 255
セグメント 487
セグメントサイズの決定 487
セグメント数 654，655，660
セグメント内の空きページ比率 488
セグメント内の空きページ比率の設定 488
セグメントの確保と解放 489
セットアップ識別子〔マルチ HiRDB〕 324
セマフォ識別子数の見積もり 746，750

## そ

相互系切り替え構成への移行〔マルチ HiRDB〕 328
挿入履歴保持列 408
総ページ数 618
総ページ数を求める計算式 598

## た

第 1 次元分割列 378
第 2 次元分割列 378
大規模なシステムの構築 320
代替可能性 232，421
多重定義 422
単一バイト文字コード 58
単一列インデクス 467
単一列分割 471
単調増加ファイル 9

## ち

中国語漢字コード（GB18030） 58

抽象データ型 420
抽象データ型（SGMLTEXT 型）を定義した表の作成方法 212
抽象データ型（XML 型）を定義した表の作成方法 215
抽象データ型の定義 230
抽象データ型のナル値 232
抽象データ型の列のデータ長の求め方 605
抽象データ型列構成基表 233
抽象データ型を含む表 420
重複キーインデクスに関する質問 811

## つ

通知できる HiRDB のイベント 267

## て

定義情報の保存と読み込み〔簡易セットアップツール〕 72
定義の更新〔簡易セットアップツール〕 72
ディクショナリサーバが使用する共用メモリ 572
ディクショナリサーバ定義の作成 140
ディクショナリ搬出入ユティリティ（pdexp）実行時のファイルの容量 719
ディクショナリ搬出入ユティリティ（pdexp）実行時のメモリ所要量 737
ディスク容量の確認〔インストール〕 49
ディレクトリサーバ連携機能 4
データ操作と整合性〔検査制約〕 455
データ操作と整合性〔参照制約〕 443
データ長一覧 603
データ長一覧〔可変長文字列型〕 605
データ長一覧〔繰返し列〕 607
データ長一覧〔抽象データ型〕 606
データディクショナリ LOB 用 RD エリアの作成 177
データディクショナリ LOB 用 RD エリアの総ページ数 654，655
データディクショナリ LOB 用 RD エリアの配置〔HiRDB/ シングルサーバ〕 288
データディクショナリ LOB 用 RD エリアの配置〔HiRDB/ パラレルサーバ〕 317
データディクショナリ LOB 用 RD エリアのページ長 654，655
データディクショナリ LOB 用 RD エリアの容量の見積もり 654
データディクショナリ用RDエリアの容量の見積もり 618
データディレクトリ用 RD エリアの総ページ数 653
データディレクトリ用 RD エリアのページ長 653
データディレクトリ用 RD エリアの容量の見積もり 653
データの格納状態の確認 215，236
データの変換方式の設定〔データベースの作成〕 199
データページ 336
データベース回復ユティリティ（pdrstr）実行時のメモリ所要量 735
データベース構成変更ユティリティ 170
データベース構成変更ユティリティ（pdmod）実行時のメモリ所要量 730
データベース再編成ユティリティ（pdrorg）実行時のファイルの容量 710
データベース再編成ユティリティ（pdrorg）実行時のメモリ所要量 728
データベース作成ユティリティ 199
データベース作成ユティリティ（pdload）実行時のファイルの容量 708
データベース作成ユティリティ（pdload）実行時のメモリ所要量 725
データベース状態解析ユティリティ（pddbst）実行時のファイルの容量 716
データベース状態解析ユティリティ（pddbst）実行時のメモリ所要量 732
データベース初期設定ユティリティ 165
データベース初期設定ユティリティ（pdinit）実行時のメモリ所要量 724
データベース定義ユティリティ（pddef）がエラーになる場合 815
データベース定義ユティリティ（pddef）実行時のメモリ所要量 725
データベース定義ユティリティ（pddef）の実行に関する質問 810
データベースに関する最小値 793
データベースに関する最大値 793
データベースの更新ログ取得方式 199，235
データベースの最大容量 810
データベースの作成 197
データベース複写ユティリティ（pdcopy）実行時のファイルの容量 716
データベース複写ユティリティ（pdcopy）実行時のメモリ所要量 734
データ未完状態 409
データ用グローバルバッファの割り当て 331
データ用グローバルバッファを割り当てる場合 333
データ用ローカルバッファ 350
データ用ローカルバッファの割り当て 350
データロードをする場合 764
適用機種の確認〔インストール〕 48
デスクトップヒープ指定値の見積もり 754
デファードライト処理 338

デファードライト処理の並列 WRITE 機能 339
デファードライトトリガ 338
デファードライトトリガでの更新ページの出力比率 338
デファードライトトリガの要求比率 339
デフォルトコンストラクタ関数 231，422

## と

同期点行数 203
同期点指定のデータロード 202
同期点指定のデータロード〔ユティリティ異常終了時の対処方法〕 243
統計解析ユティリティ（pdstedit）実行時のファイルの容量 714
統計解析ユティリティ（pdstedit）実行時のメモリ所要量 732
動的登録 255
動的トランザクションの登録 250
登録〔トランザクションマネジャ〕 255
登録の変更〔トランザクションマネジャ〕 262
トランザクションの移行 250，253
トランザクションの完了種別 264
トランザクションマネジャ 249
トランザクションマネジャに登録する情報 256
トランザクションマネジャへの登録 255
トランザクションマネジャへの登録の変更 262
トランザクションマネジャへの登録例 260
トリガ 382
トリガ SQL 文 382
トリガ契機となる SQL 382
トリガ動作条件の解析ツリー長 635
トリガ動作の探索条件 382
トリガの管理 386

## に

入力データファイル UOC 205

## ね

ネットワーク構成例と定義例〔FQDN〕 770
ネットワーク構成例と定義例〔マルチコネクションアドレス機能〕 772
ネットワークドライブの使用に関する質問 809

## の

ノースプリットオプションの指定 398

## は

バージョンアップ 28
バージョンアップ〔プラグイン〕 34，192
バージョンアップに失敗した場合 34
バージョンアップに必要な空き領域 28
バージョンアップ前にすること 28
バージョンダウン 36
バイナリデータ 400
パスワードの変更〔データベースの作成〕 198
バックエンドサーバが使用する共用メモリ 575
バックエンドサーバ定義の作成 140
ハッシュ関数 362
ハッシュジョイン及び副問合せのハッシュ実行時に必要なメモリ所要量 531，586
ハッシュ分割 361
ハッシュ分割表のリバランス機能 356，377
バッチファイルによる HiRDB の環境設定 822
バッチファイルによる環境設定の概要 817
バッファヒット率 335

## ひ

非 UNIQUE 属性 811
被参照表 430
被参照表と参照表間のデッドロック 434
非同期 READ 機能 337
非同期 XA 呼び出し〔X/Open XA インタフェース環境〕 250
非ナル値制約 394
非分割キーインデクス 471
ビュー表 391
ビュー表の作成 391
表及びインデクスを定義する場合 764
標準セットアップ 82
表定義時に出力されるシステムログ量 668
表データ更新時に出力されるシステムログ量 671
表の格納ページ数の計算方法 599，618
表の格納ページ数の計算例 608
表の正規化 357
表の整合性確認手順〔検査制約〕 456
表の整合性確認手順〔参照制約〕 444
表の設計 353
表の定義 233
表の定義情報〔サンプルファイル〕 761
表のマトリクス分割 378
表の横分割 361
表の横分割の形態 369
表の横分割の効果 370
表へのデータの格納 234
表を設計するときの検討項目 354

## ふ

ファイアウォールが設置されている場合の設定 778
ファイルシステムの確認〔インストール〕 51
複数接続機能〔X/Open XA インタフェース環境〕 250，257
複数列インデクス 467
複数列分割 471
部分構造インデクス（B-tree） 216
部分構造パス用解析ツリー長 622
不要な RD エリアの削除 205
プライマリキー 394
プラグインインデクス 476
プラグインインデクスの定義 214
プラグインインデクスの横分割 477
プラグインが提供する抽象データ型を定義した表の作成 212
プラグインのアンインストール 195
プラグインのインストール 186
プラグインの環境設定 185
プラグインの削除 194
プラグインの所有者 188
プラグインの登録 187
プラグインのバージョンアップ 192
プラグインをバージョンアップする場合 34
プリフェッチ機能 336，487
フレキシブルハッシュ分割 362
フレキシブルハッシュ分割及び FIX ハッシュ分割の例 368
フロータブルサーバ 376
フロータブルサーバの設置 292
フロータブルマシン 376
プロセス一覧 796
プロセス間メモリ通信用共用メモリ〔HiRDB/ シングルサーバ〕 510
プロセス間メモリ通信用共用メモリ〔HiRDB/ パラレルサーバ〕 544
プロセス固有領域〔メモリ所要量〕 509，543
ブロック転送又は配列 FETCH で必要なメモリ所要量〔HiRDB/ シングルサーバ〕 538
ブロック転送又は配列 FETCH で必要なメモリ所要量〔フロントエンドサーバ〕 594
フロントエンドサーバが使用する共用メモリ 572
フロントエンドサーバ定義の作成 139
フロントエンドサーバの複数化 292
分割キー 361
分割キーインデクス 471
分割キーの選択方法 362
分割入力データファイル 203
分割入力データファイルの作成 203
分岐行ログ量の見積もり 675
分散トランザクション処理 249

## へ

ページ 490
ページ固定 272，293
ページサーチモードの切り替え回数 500
ページ長 618
ページ長の決定 490
ページ内の未使用領域の比率 491
ページ内の未使用領域の比率の設定 491
ページ内の未使用領域の比率の求め方 492
ページの解放 493
ページの確保 493
別名〔コマンド名〕 765
ヘテロ構成 322

## ほ

ポート数 783
ポート番号 785
ポート番号の指定方法 785
ほかの製品との連携 247
ホスト名 770
ホスト名に関する質問 809
ホスト名の登録 52

## ま

マスタディレクトリ用 RD エリアの総ページ数 652
マスタディレクトリ用 RD エリアのページ長 652
マスタディレクトリ用 RD エリアの容量の見積もり 652
マトリクス分割 378
マトリクス分割表 378
マルチ HiRDB のインストール 324
マルチ HiRDB の環境設定 325
マルチ HiRDB のシステム設計 324
マルチコネクションアドレス機能〔HiRDB サーバへの接続方法〕 771
マルチスレッド対応の XA インタフェース 250
マルチスレッド用のライブラリ〔注意事項〕 264
マルチフロントエンドサーバ 292
マルチフロントエンドサーバの構成例 295
マルチフロントエンドサーバの設定 295
満杯セグメント 487

## み

未使用セグメント 487
未使用ページ 490

## め

メッセージキュー識別子数の見積もり 746，749
メッセージキューテーブル数 746，749
メモリ所要量の確認〔バージョンアップ〕 30
メモリ所要量の計算式〔HiRDB/ シングルサーバ〕 508
メモリ所要量の計算式〔HiRDB/ パラレルサーバ〕 542
メモリ所要量の見積もり方法〔HiRDB/ シングルサーバ〕 506
メモリ所要量の見積もり方法〔HiRDB/ パラレルサーバ〕 540
メモリ配置〔HiRDB/ シングルサーバ〕 506
メモリ配置〔HiRDB/ パラレルサーバ〕 540
メモリ容量の確認〔バージョンアップ〕 30

## も

文字コードの選択 58
文字集合 404
文字レパートリ 404

## ゆ

有効保証世代数〔HiRDB/ シングルサーバ〕 283
有効保証世代数〔HiRDB/ パラレルサーバ〕 311
ユーザ LOB 用 RD エリアの作成 173
ユーザ LOB 用 RD エリアの配置〔HiRDB/ シングルサーバ〕 290
ユーザ LOB 用 RD エリアの配置〔HiRDB/ パラレルサーバ〕 318
ユーザ LOB 用 RD エリアのページ長 660
ユーザ LOB 用 RD エリアの容量の見積もり 660
ユーザオウンコーディング 205
ユーザが定義した抽象データ型を定義した表の作成 230
ユーザ用 RD エリアの作成 170
ユーザ用 RD エリアの配置〔HiRDB/ シングルサーバ〕 289
ユーザ用 RD エリアの配置〔HiRDB/ パラレルサーバ〕 317
ユーザ用 RD エリアの容量の見積もり 598
ユティリティが使用する作業表用ファイルの容量 701
ユティリティ実行時のファイルの容量の見積もり 708
ユティリティ実行時のメモリ所要量の見積もり 724
ユティリティ実行時の容量の見積もり 707
ユティリティによるデータベース作成時に出力されるシステムログ量 680
ユティリティ用の HiRDB ファイルシステム領域の作成 151
ユティリティ用の HiRDB ファイルシステム領域の設計〔HiRDB/ シングルサーバの場合〕 276
ユティリティ用の HiRDB ファイルシステム領域の設計〔HiRDB/ パラレルサーバの場合〕 305
ユニークインデクス 202
ユニットコントローラが使用する共用メモリ〔HiRDB/ シングルサーバ〕 515
ユニットコントローラが使用する共用メモリ〔HiRDB/ パラレルサーバ〕 550
ユニット制御情報定義の作成〔HiRDB/ シングルサーバ〕 136
ユニット制御情報定義の作成〔HiRDB/ パラレルサーバ〕 139

## よ

横分割〔プラグインインデクス〕 477
横分割の設計 361
横分割表 361
横分割表のインデクス定義に関する質問 811
横分割表の作成 206
読み取り専用 250
予約数 499

## り

リスト用 RD エリア〔グローバルバッファの割り当て〕 332
リスト用 RD エリアの作成 181
リスト用 RD エリアの設計 494
リスト用 RD エリアの配置〔HiRDB/ シングルサーバ〕 290
リスト用 RD エリアの配置〔HiRDB/ パラレルサーバ〕 319
リスト用 RD エリアの容量の見積もり 664
リスト用 RD エリア用の HiRDB ファイルシステム領域の作成 151
リスト用 RD エリア用の HiRDB ファイルシステム領域の設計〔HiRDB/ シングルサーバ〕 277
リスト用 RD エリア用の HiRDB ファイルシステム領域の設計〔HiRDB/ パラレルサーバ〕 306
リソース数に関連する環境変数の見積もり 745
リソースマネジャ 249
リバランス機能 356，377
リバランスユティリティ（pdrbal）実行時のファイルの容量 721

## る

ルーチン 422
ループバックアドレス 68

## れ

例外リストからの削除〔Windows ファイアウォール〕 69
例外リストへの登録〔Windows ファイアウォール〕 66
レコード数の求め方〔システムログファイル〕 666
レコード数の求め方〔シンクポイントダンプファイル〕 686
レコード数の求め方〔ステータスファイル〕 687
レコード長〔システムログファイル〕 281，309，666
レジストリ LOB 用 RD エリアのページ長 663
レジストリ LOB 用 RD エリアの容量の見積もり 663
レジストリ機能の初期設定 189
レジストリ情報の登録 190
レジストリの削除 195
レジストリ用 RD エリアの容量の見積もり 661
列情報作業表の最大数の求め方 699
列情報作業表の容量の求め方 698
レプリケーション機能との連携 248
連絡用ポートのサービス名及びサービスポート番号の設定〔マルチ HiRDB〕 324
連絡用ポートのサービス名の変更〔マルチ HiRDB〕 324

## ろ

ローカルバッファ 350
ローカルバッファの設計 329
ログ取得モード 199，235
ログレスモード 200，236

## わ

ワークディスク 376
ワークファイル出力先ディレクトリの作成 59